Information Transfer

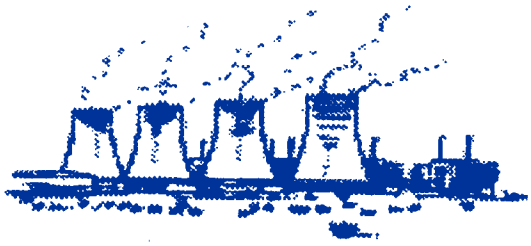
Volume 13 Number 3 Term Three October 1993

Feature Article:

Teacher Change:
Philosophy & Technolo

Reviews:

Caddman Cadet
Pacific Powernet

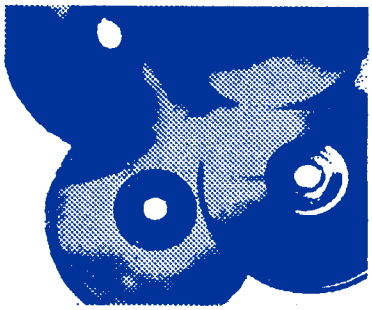


Conference Papers

PLUS. . . . .

Logo for maths tools

Using CD-ROM systems



Standard International Signs

Registered by Australia Post Publication # NBP5123

Information Transfer (IT) is the journal of the New South Wales Computer Education Group Ltd.
IT is printed at
Graham Rowell Publications,
Unit 3 Hopkins Place,
Industrial Park,
Narooma,
NSW 2546
and is published by the NSW Computer Education Group once each school term.

Correspondence relating to IT should be addressed to:

The Editor,
Information Transfer,
NSW CEG Ltd,
c/- Instructional Technology Centre,
School of Education,
Macquarie University,
SYDNEY, NSW 2109

Your contributions, whether they are letters, photographs, reviews, articles or ideas are welcome. Send material to the above address.

Editor
Neville Fraser
Publicity
Leonie Fraser

Advertisers may contact the NSW CEG by phone: (02) 805 9456 or fax: (02) 805 9453 or fax and message may be left on (02) 610 5125

ISSN 0813-4871

CONTENTS

Advertisers

Editorial

Welcome to this the third issue of IT for 1993. The more observant of you will have noted that this is a bit late in the year. Please rest assured that Issue Four is coming quite close behind, and you will get all four issues for the year!

I think everybody has had a busy year, and the workload of teachers and lecturers has increased disproportionately to any increase in support. The expectations of implementing new syllabi*, quality assurance, and the spectre of "structural change", as well as a widespread 'flu epidemic, has made for a difficult time.

This is an issue of IT with a bit for everybody, I feel. We again include a feature article, this time letting you consider your "philosophy" in computer education. This is a point that I feel strongly about, inflicting a lecture to my trainee teachers each year! See how it effects you. . .

What about some high level Logo programming? There is an article from the LOGO '93 Conference in Melbourne. No doubt it will have maths teachers getting excited!

Not all the good material from the National Conference was able to be included in the Proceedings or Journal, so we will be printing some of it over the coming months. This issue there is our feature article, plus items on CAL and an alternative to traditional programming. I hope you get something out of them.

Finally, there are our reviews. I always like to get reviews from you, as they form a very useful link between what the suppliers think is good for you and what academics think you should be able to do. If you've tested out some software or hardware, let us know about it. Even a single page is fine.

One of our reviews this issue is very special. I was lucky enough to be at the launch of Powernet, which has the potential of being a very powerful resource. I do have some reservations at present, and I admit that the style of the launch left a lot to be desired as far as I'm concerned, but I'll not shoot the thing down before it starts.

The Department is putting together a Computing Studies Intensive Methodology Course which will be piloted early next year. I have been part of the design team working on methodology. It is good to see the various teachers, consultants and lecturers getting together nut out a course like this- its's not before time!

Next issue? How about some video reviews, more NatCon papers, more Logo, and some Christmas treats? See you then!

Neville

TEACHER CHANGE: PHILOSOPHY AND TECHNOLOGY

Helen McDonald
Methodist Ladies' College, Melbourne

ABSTRACT:

Computers and their related technology may be relatively new to the educational scene, but the process of change is not. Persuading people to change is rarely an easy task. Change can seem threatening, it can bring insecurity, anxiety and a great deal of hard work. The process of change is personal and time-consuming, requiring commitment and support. Thus to embark on what may be a difficult process (and perhaps a stimulating and exciting one) requires a conviction that the journey will be worth it - a belief that the outcomes are worth achieving.

This paper looks at the relevance of a firm philosophical foundation as a basis for technological change. It also notes the importance of action-research in re-assessing the purposes, beliefs and practices of teachers. Much of the research referred to relates to a case study of Methodist Ladies' College, Melbourne, where a program based on independent learning and utilizing laptop computers has been operating since 1990.

Change can be achieved on many levels, from a superficial adoption of a few visible facets, to a complete assimilation of the change which challenges and alters old structures and existing beliefs. Real change must go below the surface, beneath the adoption of hardware and software, beyond the acquisition of new skills. For teachers it means looking at learning and re-examining its processes.

> *I need three more days in the cycle to be able to get through all of this! I feel like I can never do anything properly: I'm just skimming over the surface. I haven't got time to read what I need to read and I haven't got time to digest it. Look at this pile of work I have to get through!*
>
> *Please - no more change! We haven't got through the last one yet. I need to have time to think about what I am doing instead of changing it every five minutes!*
>
> *(M.L.C. teacher Feb. '93)*

To many teachers, frequent change is not an attractive proposition. It is too demanding, too exhausting and often doesn't seem worth the effort. Change takes time and it often requires the relinquishing of control. No wonder that some teachers prefer to "cultivate their gardens", changing things they can control - small pieces of the syllabus; additions to a program - safe innovations that are manageable. (Huberman, 1992, p.5) By so doing they avoid the roller-coaster ride of major change which may exhilarate one minute, and plunge innovators to the depths of despair the next.

At Methodist Ladies' College in Melbourne, the roller coaster ride is mandatory for all staff. The climate at this school is demanding but it also encourages risk-taking, supports innovators, invites experimentation, copes with mistakes/ problems and celebrates achievement. Envisioned as "a learning place for teachers as well as students" (Loader, 1993, p.6), M.L.C. seems to have developed an institutional capacity to innovate. Coping with change appears to improve with practice and many of the existing staff are well-versed in the process. The administration both supports and encourages (or some may say pressures) active participation in innovative programs and has established formal networks to facilitate change. That is not to say that all change at M.L.C. is successful, nor does it come without the associated "pain and disorder" which are "healthy signs" of school improvement. (Huberman, 1992, p.6)

The particular change upon which this paper focuses is one of many occurring at M.L.C. and should be seen in that context. The innovation centres upon constructionist thinking and the encouraging of effective independent learning at Junior Secondary level (Years 7 and 8), although both concepts are not exclusive to this section of

the school. Linked significantly to this philosophical base is the integration of laptop computers and the use of LogoWriter.

The Principal, David Loader, speaks enthusiastically of M.L.C's move towards a more constructionist approach, which centres upon the individual student's learning:

> *This approach is based upon Piaget's "constructivism" where knowledge is "built by the learner, not supplied by the teacher." This idea has further been extended by Seymour Papert to "constructionism" which includes "the further idea that this happens especially felicitously when the learner is engaged in the construction of something external or at least shareable ... a sand castle, a machine, a computer program, a book." (Polin, 1990. p.6)*
>
> *The idea of knowledge being constructed by the student shows appropriate respect for the intellect of the learner and reflects the subjective, and evolutionary view taken of knowledge. The student, in such a view, is not a passive recipient of data but a constructionist trying to understand her world, having meaningful experiences, making personally significant connections, developing mental models, collaborating with others in an enriched teacher supported social setting.*
>
> *(Loader, D. 1993, p.4)*

peers is more "inter-dependent" and co-operative. Computers and other forms of technology, were seen as having strong potential in assisting the implementation of these beliefs and so were introduced concurrently. In 1989 a pilot class was taught by a team of three, following the ethos of independent learning. Four computers were located in the classroom as a resource and the program LogoWriter, was used.

In 1990, four Year 7 classes took the innovation on board, with one class using laptop computers. School policy requested that each student in this class own their own laptop computer at an initial cost of approximately $1,500. The success of the project and subsequent parent demand saw nine out of 10 classes in 1991 and 1992 using laptops, while the remaining classes had daily access to the computer network. In 1993, all year 7 students use their own laptop computer. LogoWriter has been integrated into English and the Humanities (History, Biblical Studies and Geography), Mathematics, Science. and some L.O.T.E.. In teacher terms, the team expanded each year, attracting like-minded staff until the change became policy and all Junior Secondary teachers became involved.

The idea of knowledge being constructed by the student shows appropriate respect for the intellect of the learner

In M.L.C.'s Junior Secondary School, this view became woven into the philosophy of independent learning. The aim was to create a learning environment which highlighted individual differences rather than suppressing them. Students were encouraged to work effectively without constant direct instruction. They could set their own working pace, have some flexibility in areas of study, and pursue options which motivated them - all within a framework provided by the teacher. "Learning experiences" were emphasized rather than "teaching programs" as students were encouraged to take responsibility for their own learning. Their role became more active: observing, participating, working together, constructing and reflecting . In fact, "independent" learning is perhaps, a misnomer as the relationship with teacher and

Many people both within and outside the M.L.C. community tended to view the highly visible laptops as "the change". Some labelled it as a "gimmick" or an "enrolment booster" - part of a technological bandwagon that would provide a clear advantage to those who were a part of it. However, those who initiated the program would argue that the laptops were introduced only to facilitate and enhance the possibilities created by independent learning. In many ways, laptops were not the change: the change was in thinking, in learning and in teaching. The technology supported the change as a rich resource.
As one M.L.C. teacher stated:

> *You can't put a kid in a classroom with a laptop and LogoWriter and expect miracles to happen. The computer itself doesn't do a thing for you. It needs someone to make it happen, someone to*

> *give you ideas. It needs a philosophy of how and why we are doing it - it needs an aim. It needs people who know how to go about achieving it.*
> *(Interview: 2/4/92)*

And so the teachers who worked in the Junior Secondary School had to assimilate the views of constructionism and independent learning with their own personal educational outlook. For a few - particularly the initiators who were involved in the pilot program - the philosophy, the technology and their own educational outlooks were compatible from the outset. For others, the task was not so easy.

Some accepted the philosophy, but shied away from the laptops and the technology entailed. Others embraced the computers, seeing the technology in terms of professional development and as a means of enhancing their teaching. A third group wished only to continue "cultivating their gardens", and tended to accept the trappings of the change without significantly altering practice.

However, there does seem to be a point where these paths can merge. Once teachers have come to grips with the more operational aspects of the change (whether it be computers or independent learning) , they begin to build the mental models necessary to provide inroads into the other aspect of the change. For example, the teacher who embraces computers soon comes to realize that students can and do work more independently with the laptop. The flexibility provided in recording, generating and presenting information allows students more scope. The medium also seems to encourage more frequent interaction among peers - such as group problem solving.

Once teachers have come to grips with the more operational aspects of the change (whether it be computers or independent learning) , they begin to build the mental models necessary to provide inroads into the other aspect of the change.

Whether intentionally or not, the teacher's role in the classroom changes. No longer the expert, the teacher must learn to facilitate. No longer the instructor, the teacher guides. The philosophy of independent learning becomes relevant and the reality of the classroom makes the connection between beliefs and practice.

Alternatively, for those who believed in the philosophy, the value of laptops and LogoWriter quickly becomes apparent. As students use LogoWriter in most subjects, they begin to utilize its functions across subject boundaries. Teachers do not need a great deal of technical expertise to allow the laptop to be used as a tool in their classes. They do not need to be master programmers to mark a piece of work on disk. As they see the technology assisting students with organization, drafting and problem solving, it quickly becomes apparent that its link with independent learning is a valid one.

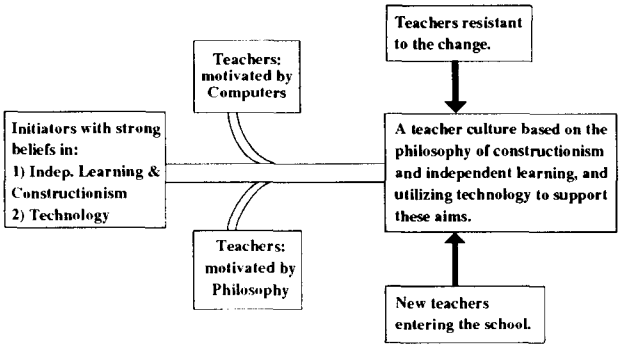
The third group, teachers who are resistant to the change, can be quite successful in limiting changes to practice but only for a short time. As students become skilled with the technology, they begin to use it in all subjects. A teacher who does not encourage computer-use soon becomes visible and has to deal with a situation where students expect consistency of approach both in classroom operation and learning style. Students do not react well to a student-centred approach in five classes and the traditional teacher-centred approach in one.

This reticent teacher group, although not experiencing the changes first hand, can not fail to observe the educational outcomes as they become apparent. Subject meetings soon include frequent demonstrations of successful programs generated by colleagues; the staffroom conversations introduce the jargon associated with both constructionism and computers and the administration begins to insist that all reports are generated using a computer. As the culture of the school alters, those who avoided the change feel increasingly "left behind".

To meet the needs of all teachers, a support system must continue to provide opportunities for staff to tap into a learning network which allows participants to obtain new knowledge and skills as the need arises. Gary Stager (Interview: 1991) described it as "casting small nets" - to encourage

teachers, at the point where they feel ready, to "have a go" and to be part of the emerging culture. New staff entering the school must also be catered for, as they bring with them knowledge and skills which may further build the culture, but may also need support to acquire the skills necessary to teach in the established program.

This process where the divergent paths tend to converge may be illustrated diagrammatically:



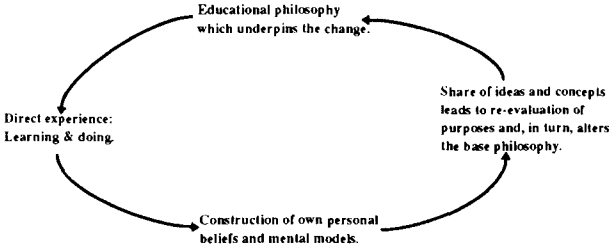
Hence, although the school succeeded in providing a solid and appropriate philosophical foundation for the introduction of technology, teachers often had to work with laptops and within the new structures - forging their own links and meanings. As Fullan (1992: p. 25) states " It seems that most people do not discover new understandings until they have delved into something. In many cases, changes in behaviour precede rather than follow changes in beliefs." Some teachers also need to see outcomes emerging before committing themselves personally.

The importance of "beliefs" and "understandings" is central to the success of a change process. As Michael Fullan states:

> *All substantial innovations have underlying beliefs, rationales or philosophies...(Those involved will) have to come to understand and believe in the new assumptions and ideas that underline that reform. 9*
>
> *Fullan, M.G 1991, p.2*

However, the point in the change process where that meaning is obtained is not set or fixed. If a school's philosophy is based on constructivism, then teachers too must be encouraged to build their own mental models and develop their own meanings and beliefs. This includes the active provision of experiences which confront theory and allow for the evaluation and re-evaluation of aims and purposes. As teachers conduct and share their own action-research, meanings can be exchanged and a culture extended which includes and accepts the experiences and beliefs of all participants.

In such a learning environment one needs to see learning philosophy as fluid and as changing. Just as teachers are influenced by the provision of a set of beliefs and concepts, proposed as a foundation for teaching and learning, so does their experience influence and alter the educational philosophy. Action-research may result in the extension of ideas, changes in emphases, the rejection of some concepts and the integration of others. And so the cycle of theory and practice continues, constantly evolving.



Fullan and Miles (1991) describe "change" as "a process of coming to grips with new personal meaning ... a learning process." It requires the participants to alter their purposes, develop commitment to new ways and to "unlearn old beliefs and behaviours." (Fullan and Miles, 1991. pp10-12) As such it requires great effort from individuals who must cope with anxiety, difficulties and uncertainty as they seek , not only to introduce an innovation but to change the culture of their schools.

To embark on major change - to take on board new ideas and skills is demanding. Change takes effort and a great deal of commitment. Anxiety, uncertainty, frustration, overload, lack of skill, complexities, incompatibility: all were experienced in the M.L.C. study. The process of change is not easy as the comments below indicate:

> *It was the most frustrating thing I've ever been through. There was a lot of expectation placed on teachers ... the approach was made compulsory, but where do you find the time to build new skills? It was assumed we would do it, assumed that you would understand and be able to cope -*

but some people did not understand and some didn't cope and some didn't really try.
(Interview 6/12/91)

It's exciting, but you wonder how long people can maintain this enthusiasm and commitment... you wonder how long teachers can keep up the pace and the pressure. The environment for the kids is fantastic - they have exciting experiences, computer experiences and all that. But the teacher workload is very, very high.
(Interview 19/3/92)

If the process of change is to be experienced as accomplishment rather than frustration, a strong sense of purpose needs to be accompanied by many practical considerations.

As Linda Darling-Hammond writes:

...the process of change is slow and difficult. It requires perseverance, and it requires investments in those things that allow teachers, as change-agents, to grapple with the transformations of ideas and behaviour: time for learning about, looking at, discussing, struggling with, trying out, constructing, and reconstructing new ways of thinking and teaching.
L. Darling-Hammond 1990 p.240

In practical terms M.L.C. did much to support staff with workshops, rebates for computer purchase, educational courses, classroom support, access to information and a network of people who could provide assistance.
Most activities were voluntary and repeated, thus allowing staff to join in when the experience would be most meaningful and/or convenient.

One of the most worthwhile professional development options was the running of residential workshops, which took staff away from their usual commitments and provided the time "to play" with LogoWriter and discuss philosophies of learning in a supportive environment. As well as formal structures, personal networks emerged, through which teachers could obtain information without feeling anxious or incompetent.

Support must be planned for, but it must be flexible enough to serve a variety of needs at the same time. As needs change and new demands are made, so support structures must adapt, for it is only after attempts are made to implement change, that we come to understand more clearly what we need to know.

Teachers, like all learners, should be given the opportunity to make sense of change. Whether a philosophy of learning and teaching, or a technology- based innovation, staff must be allowed to experiment, to make mistakes, to explore and to discover. Structures need to be developed which allow for the free exchange of ideas and experiences in a supportive environment.

The initial motivation to participate in change may come from many directions, but a climate should exist where participants can feel that their involvement in the change has purpose and will provide benefits.

To ensure relevance and a valid framework for change, the existence of a clear philosophy needs to be shared with those involved. Nor should the change or the philosophy be viewed as a static package which is to be "learned" or implemented. Meanings and beliefs need to emerge through experience and practice: they need to be shared and they need to be challenged.

Encouragement can take many forms but, as in a financial transaction, people need to be able to see a definite return for time and effort invested. Learning outcomes need to be monitored and recorded, changes in relationships and roles need to be discussed - and adaptation must occur as needs dictate. In this regard, the role of action-research is critical to ensure that theory and classroom reality support each other.

Helen McDonald, a secondary English teacher and currently a Ph.D. student from Monash University, has been conducting research at Methodist Ladies' College, Melbourne in the field of educational change.

REFERENCES:

Darling-Hammond, L. (1990) **Instructional Policy Into Practice: The power of the bottom over the top**, *Educational Evaluation and Policy Analysis*: vol. 12 no 3 Fall 1990 A.E.R.A. pp. 233-252.

Fullan, M. (1991) **Overcoming Barriers to Educational Change**. University of Toronto: *Paper commissioned by the Office of the Under Secretary of the U.S. Dept. of Education.*

Fullan, M. (1992) **Successful School Improvement**. Open University Press, Philadelphia.

Fullan, M.G. and Miles, M.B. (August 1991) **Getting Educational Reform Right: What works and what doesn't**. *Submitted to Phi Delta Kappan.*

Huberman, M. (1992) **Critical introduction to Fullan, M. (1992) Successful School Improvement**. Open University Press, Philadelphia. pp. 1-20.

Loader, D. (1993) **Reconstructing an Australian School**. *submitted to The Computing Teacher.*

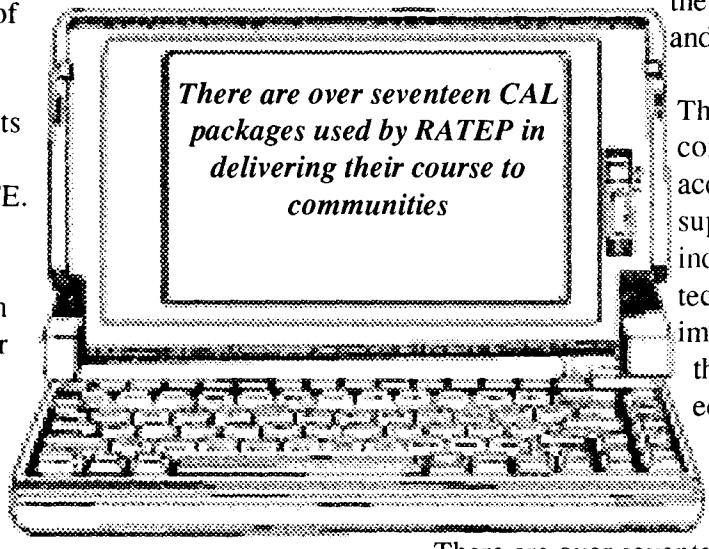
Polin, Linda. (1990) **What's hot and what's not**. The Computing Teacher. August/September.

CAL: What's in a Word?

Melissa Nursey-Bray & Karen Reys
Cairns College of TAFE

INTRODUCTION

The relationship between computer technology and education has developed to enhance delivery and learning programs. Nowhere is this most evident than in the field of remote and distance education. One particular development in this area is the use of CAL. (Computer Assisted Learning) packages. The use of these programs has advantages and disadvantages. This paper will discuss some of these within the context of the Community Ranger Training program and Remote Area Teacher Education Project (RATEP) of the Cairns College of TAFE where CAL has been used extensively. Discussion will first centre on the use of CAL by RATEP, concentrating on how CAL works, and how its use developed at the Cairns College of TAFE. The second part of the paper will examine aspects of CALs' use in the field with particular reference to the Community Ranger Training program.



RATEP : REMOTE AREA TEACHER EDUCATION PROJECT

RATEP provides teacher education by remote delivery to Aborigines and Torres Strait Islanders living on remote communities in the Cape York and Gulf regions. It offers at seven sites (ten by the end of 1993) an Associate Diploma of Education (Community Teaching). Students can then continue through RATEP to undertake the second year of the Diploma of Teaching with JCU. RATEP is a co -operative effort between Cairns College Of TAFE, James Cook University and the Queensland Open Learning Centre Network (QOLCN).

The program evolved in response to the need to enable Aboriginal and Torres Strait Islander people to attain the same qualifications as non - indigenous teachers. It thus performs a vital function in allowing students to attain teaching qualifications without having to leave their communities. It also allows Aboriginal and Torres Strait Islander participation in decision making through their representation on the Curriculum Development Advisory Committee, the Management Committee and the Reference Group.

The use of CAL within this context provides equity and access in education. It also supports self paced and independent learning techniques. These are important both in addressing the problems of remote area education delivery and through enhancing levels of self esteem.

There are over seventeen CAL packages used by RATEP in delivering their course to communities. These include literacy, and teaching packages which use community examples and relevant contexts to develop concepts. RATEP has also developed information packages on its course and one which explains CAL and its functions.

HOW CAL WORKS

CAL packages are delivered to the sites on 44 megabyte removable cartridges (PLI/Syquest) and then loaded onto a hard disk through a PLI drive. While still in use, the overall aim of RATEP is to

offer CAL through use of CD - ROM, which is easier to develop as CAL can then be directly programmed onto the CD. Benefits of this will include greater storage capacity, and therefore highlight the benefits of CAL. These include greater video, graphics and extra sound effects.

CAL is created by computer programmers who develop the packages from the material given to them from course writers. Authorware Professional; which is a multimedia software package using Macintosh based hardware, is used as the foundation for development of CAL. A CD Rom is used to add music to the packages due to the quality of sound that this offers. A colour scanner is used to include photographs, and colour graphics and designs for the package. However, colour and sound take up much space and thus their potential has not yet been realised as student machines have lesser megabytes available.

Macromind Director while used as an animation tool is also not used extensively as it is not possible to use Macromind Accelerator in conjunction with Authorware. The inclusion of video is likely to be an alternative to this process. Quicktime and a RasterOps Board provide programmers with the opportunity to incorporate short video segments within the CAL package. The most recent RATEP package has been able through the use of Quicktime video, to use action shots to teach court procedures and other things.

CAL is simple to operate once the user is familiar with its mechanics ; students are asked to push a button, use the mouse to drag items over, type in text, use the menu's or listen to and observe the information presented. The student can then respond, try again, go back to start, quit or be corrected. A series of encouraging icons and noises are used to facilitate student interaction. These include clapping noises, cheers, happy music, or verbal/written messages such as "Very Good!"

CAL: CURRICULUM DEVELOPMENT AND DESIGN

CAL is used to respond to the educational and developmental needs of the students and the learning objectives aspired to by the course writer. Ideally, this requires a good interaction between course writer and programmer to ensure the succinctness of such response. Within RATEP this has not been a problem as there is close communication between the two, and the course writers have been aware of the needs of students on community. However, where these needs are not perceived, (as is sometimes the case if a course writer is external to the program), the CAL design accordingly, may not fully explore its potential for the subject, or remote educational requirements.

An important aim of CAL is to encourage self paced learning. Course development and design therefore is in line with this objective. The interactions that can be developed are many, and if designed carefully, curriculum objectives can be met through them, without putting pressure on the students. RATEP has put together a course writing outline for coursewriters to facilitate this process.

By and large a serendipitous approach to design has been adopted where; time constraints notwithstanding, programmer and course writer together create an effective program, with adequate feedback choices and covering the main points. This approach has ensured a creative outline for each package, and allowed a freshness to each amalgam. The flexibility in presentation that CAL offers, is one of its key attractions, especially as there are many avenues to yet be explored.

Characteristic of CAL within this context is its ability to present different learning pathways. These may be linear, cyclical, random or according to student choice. For example, it is possible for the student to enter the package and continue in a linear fashion until the end. Or it is possible to choose between a series of options, and click on one of those. These options may develop in a cyclical fashion ie. coming back to the same point at the end, or may just finish and allow the student to choose the next one. The parameters of these pathways, especially cyclical have not as yet been explored to their full potential in terms of relevant and innovative learning strategies. Nonetheless, CAL compares favourably to other educational packages such as

Hypercard that are limited in the pathways they present, and thus their interactions.

It is clear then that CAL offers various opportunities for educational development previously ill utilised. Used in the field, CAL has differing levels of success, and it is to its problems and advantages in remote education in particular that we now turn. The Community Ranger Training program is a good case in point.

COMMUNITY RANGER TRAINING PROGRAM

The Community Ranger Training program delivers a Certificate and Advanced Certificate in Cultural and Natural Resource Management, and an Associate Diploma in Applied Science. The course is delivered to over seventeen communities. These include the Gulf communities, Northern Peninsula communities, East and Southern coast and Wet Tropics communities.

The Diploma consists of twenty modules and has been developed in response to requests by Aboriginal Community Councils through the Aboriginal Co -ordinating Council. These communities identified an urgent need to become actively involved in land management and site protection. Community Councils have since undertaken to employ trainee rangers. This course offers trainee rangers the opportunities to gain many skill appropriate to their need.

The Associate Diploma of Applied Science although one of three levels of attainment, is the only one in which CAL. has so far been used. Anthropology and Archaeology , Politics and Change , Introduction to Management, Australian Plants and Animals. and wildlife Management are all subjects that have utilised CAL packages to varying degrees. This list illustrates that CAL packages can be applied in a variety of contexts.

CAL IN THE COMMUNITY

There have been many positive reactions to the CAL. package from the students. This has been because the interplay between the screen and the person doing that package is quite personal. The CAL package does not rush one, and it does not put pressure on the student.
In some cases this has been good when a student has a learning disability and needs to take time over his/her answer. The student is the one the computer will interact with. Therefore if a student types in a right or good answer the computer will reply with clapping noises, visual ticks and a "Very Good!" message. Moreover the student can replay screens if the exercise was not understood.

The CAL can be stopped at any time and taken up again at the students leisure. This is highly advantageous to learning, especially in remote areas and on community when there are so often other things which take priority. CAL offers a flexibility in this respect that a travelling co - ordinator cannot accommodate - having to come and go from that community at a given time. The student then has the opportunity to absorb information in his/her own time but also therefore to realise what is not understood and ask for clarification after that.

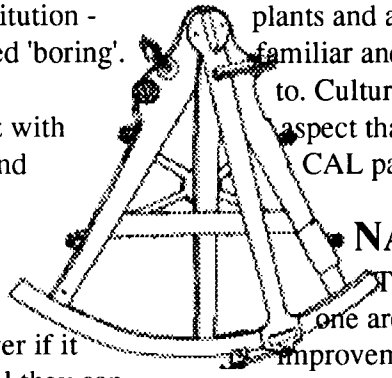
PRESENTATION TECHNIQUES

One factor important to discussing the success of CAL has been the perception that it is an impersonal mode of delivery and presentation. Some students, indeed most people have an opinion about computers; many having a psychological block perceiving them as isolating and and unfriendly. This is not a response to CAL but the computer. One way of reducing this fear is to put students through shorter packages to start off with. Another useful technique is to sit the whole group together to go through it the first time, selecting the most fearless student to operate it! Moreover, the ability of CAL to respond by name to the person operating it creates a more personal learning atmosphere for the student.

The CAL. package is advantaged by having sound and colour. It provides a vibrancy to learning that is encouraging. To some extent in fact, the packages have also helped to breakdown some of the misconceptions about computers. One experiment that worked particularly well was in using examples, pictures and photographs of the rangers themselves which made the CAL. exercise more humorous generally and therefore the issues absorbed more succinctly. Musical

noises such as the didgeridoo, animal noises (in the CAL Mammals) and computer music have added to this dimension of CAL and its success.

CAL. packages promote a creative approach to information and presentation. Video, written exercises, tables, graphs, interactive diagrams, photographs, maps, and audio narrative accompanied by words or pictures allows some information that is often hard to absorb normally, to become more readily conveyed. For example one CAL. package was describing the various structures of parliament and the constitution - information usually classically labelled 'boring'. In this case the package presented the information as if in a game show quiz with faces of ministers and a whos who? and whats what? exercise.



The length of the CAL is also important. A CAL over forty minutes is probably too long. However if it is cut up into sections the students feel they can finish one section and then move on.

COURSE CONTENT

Of course the content of the CAL. package is an influencing factor in its success. Some CAL packages have been more successful than others partly due to its content. For example, even though the CAL on parliament is presented in a most innovative way the content did not please as much as the CAL on Anthropology and Archaeology and Australian Plants and Animals.

The choice of suitable material by the course writer concerned is of importance here. It is often a matter of taste and inclination that provides the variation in course content and approach in CAL. The awareness of such course writers to both the need of the students and how they perceive they should be taught has also caused variation in the packages. Some packages are more accessible than others. The incorporation of cultural and/or community content is another factor.

CAL AND CULTURE

The nature of the ranger program is such that the cultural context of any given subject is very important. The CAL. packages that have been most popular have taken note of the cultural/ community context. The Management CAL. is one such example. It is not easy however to accommodate this aspect as there are so many different communities delivered to and so many different cultures. CAL.packages have not so far been developed for the Traditional Management Courses - CAL. does not have the ability except at a very basic level to deal with these issues nor is it always appropriate. CAL. such as with the Australian Plants and Animals draw their examples from the regions or countries of those communities; in this instance using pictures of plants and animals with which the students were familiar and could relate the biological processes to. Cultural content and relevance is certainly an aspect that needs close attention in all future CAL packages for the course.

NAVIGATION SYSTEMS

The technical smoothness of CAL is one area that has changed and still needs improvement. The navigation system in particular has become increasingly more sophisticated since the first packages developed. The first packages were particularly hampered in making the student redo the whole package up until the point at which he/she finished last time. This still happens but packages are now cut into sections and therefore the student can go to an individual section. Nonetheless this detail means there is undoubtedly pressure to finish and promotes a tendency not to return to the package if an interruption occurs.

Moreover, packages are still a little slow in terms of dragging answers, responding to students answers and continuing to the next screen. There is a lag time and this can prove boring and counter the effect of the audio visuals and music.

Students often cannot simply go back to the next screen to examine or look at a diagram - the exercises of which they have just completed. If a student wishes to re - examine this information they must do the entire exercise again. This usually means they will not go back.

Learning ways to improve in this regard is a process of trial and error. However as a result, CAL navigation systems undergo constant change and improvisation, becoming smoother each time to operate.

LOADING AND PRINTING CAL

Crucial to the smooth operation of CAL. packages is their loading. This has been a big problem in many communities in that often tutors or students are not aware of how to load a package. If in the process something goes wrong, the CAL often will not work. This is not a major problem in that it is easily fixed with training. However it must be realised that not all communities have access to computers, except perhaps through their councils, and thus familiarity with it is limited, and potential for confusion greater.

One technical aspect that has been particularly successful is that there is now an option to print every screen. This means that if a student wishes to redo keep any particular information it can be obtained in hard copy. Printing also opens up options for assessment within each subject.

ASSESSMENT

With regard to assessment there is a great deal of difficulty in defining CAL's role. In the Ranger program so far, the CAL. exercise has been used as a supplement to the work in the course or workbook, or presenting new information. Some questions in the workbook that have been assessable have relied on the students preview of CAL. For example in the Politics and Change workbook an exercise asks; Explain the Constitution. How many levels of parliament are there? The differing conception of time between aboriginal and western peoples formed the basis of a different discussion in the Management workbook - but also relied on a preview of the CAL. on time management.

The Australian Plants and Animals CAL is different in that having the print option has made it possible to print out an assessable exercise. One example is where the student is asked to explain how an oxygen molecule in the atmosphere becomes converted to use in the hind leg of a running dingo. Having typed the answer in, the student has to print it out and attach to the workbook. Or the students are asked to label the parts of a flower drawn on the screen and then print the diagram out. This has been quite a fruitful exercise as students appear much keener to answer the CAL in order to get a printout.

It is not possible however to translate all of the assessment through CAL as it defeats the purpose of it, which is to advocate self paced learning. Learning will not be self paced if it is under the pressure of a constant assessment scenario. Therefore in the ranger program while certainly possible to use CAL. as a workbook supplement it cannot replace it.

THE FUTURE OF CAL

Which brings us to the question - what is CAL's future role? CAL. is certainly capable of high sophistication in delivery and educational learning strategies. However, it is not viable that CAL. become the only option for Remote delivery education programs. The human element is a vital one and the human mind so immeasurably more complex than even CAL. that it takes a responsive mind to perceive where students are having difficulties in comprehending. The unique personality of each individual means that although CAL. is diverse in approach it is not diverse in response. It is response that is crucial to facilitating educational awareness and thought.

This is where CAL has another problem - in the way it elicits responses. CAL is presently structured to ask for certain answers to certain questions. If they do not come up the student gets a message saying "Try Again". This promotes the idea that there are wrong answers and right answers whereas some wrong answers may in fact reflect a highly creative and reflective response to the question - a fact that can be recognised by a teacher, even though the answer shown may strictly speaking be incorrect. Or even, while the answer may be wrong it may show incredible improvement on behalf of that particular student. In this instance CAL. does not have the ability to cater to the individual needs of the students or anticipate them.

This approach apart from anything else could facilitate that type of education structure that examines questions like Two plus two equals ? rather than Why does two plus two equal four ? This stilts the development of creative thought patterns and learning can become narrow in conception. This is where the collaboration between course writer and programmer is crucial to ensure there is a mutual understanding of the possibilities of CAL.

CONCLUSION

CAL. then for all its advantages is best used as an accessory and supplement to delivery, even in remote areas. The teleconference still has a more personal touch, as do co-ordinators or tutors. CAL. is however invaluable in maintaining student interest and humour, with the printing option providing ways of obtaining graphic information immediately. CAL also is, without question providing some of the most innovative advances in education and delivery strategies, and its future potential and directions can be looked forward to.

ACKNOWLEDGEMENTS

Gotts, A (1993). Using Quicktime to Teach Literacy: Instructional Design ideas for low cost users. Cairns College Of TAFE.

Gotts, A. Kruger,L.Colmer,D. Makray, J. and Hill, R. (1992) The Remote Area Teacher education Project (RATEP): An initiative in Interactive Multimedia Technology. Cairns College of TAFE.

- JUNE (1993). Discussions with Rosemary Hill, Alison Gotts, members of the Computer Programming Team for Ranger and RATEP, and Co - ordinators from the Ranger Training Program.

To you, teaching is second nature.
(Computing probably isn't)

You're already a trained teacher.

So why spend months learning a computer system? With an Apple Macintosh, you won't have to. We designed it with productive professionals in mind.

Our computers are so easy to use, you can start work straight away using the Macintosh for what you do best - your job.

Your Macintosh will make lesson preparation easy and enjoyable. And you'll be saving time with the many teacher productivity programs available - programs that let you produce standard program formats each term, maintain reports on your students and weight and scale marks automatically.

Suddenly, preparing the school newsletter, notices and take-home notes for parents is half the chore it used to be. The Macintosh has many simple publishing programs to make your work easier and the end results will look great.

Storing work on the computer disk means that next time you need a similar hand-out, it's just a matter of altering dates and venues, without reworking the whole concept. With your Macintosh you'll find regular work takes a few minutes to re-do instead of hours to do over again, giving you more time with your family and friends.

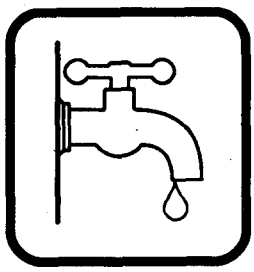
So if you're the kind of teacher who loves teaching, but doesn't want to know about computing, the friendly Macintosh is the computer you've been looking for.

For more information, contact Chris at 008 025 355 for the location of your nearest Apple Education Reseller.

Dedicated to learning.

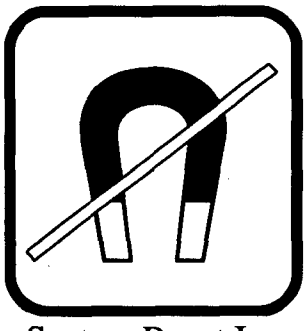
Just about every clip art collection in existence has a selection of "International Signs", but often these signs are without any explanation of their meaning.

As we in Australia lead sheltered lives, here are some of the international signs and their meanings:



Information Flow Point

Virus Protection Installed



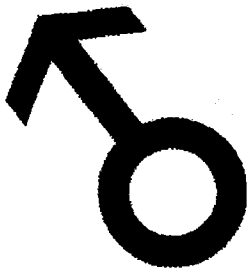
System Reset In Process

User Group Meets Here

Wait Here For Printer To Finish

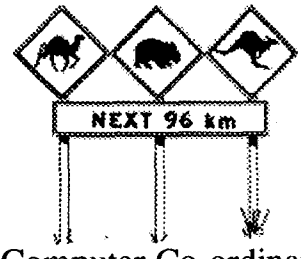
Technician Available Immediately

Old Software Prohibited



Computers May Be Thrown From This Point

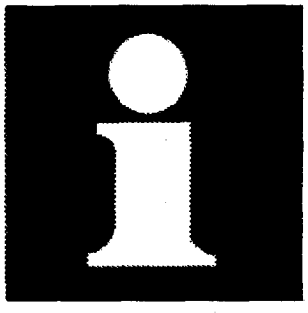
Newton Powerpoint Ahead



Computer Co-ordinato Next 96km

Attempting To Carry This Computer Will Break Your Legs

Help Is Just A Phone Call Away



I've Left Your Report Printing Over There. . .

Plugging Leads Into Left End of CPU Could Cause Fires

Can You Write Me Some Tools That... *

Clive Lynn

***Reprinted from the LOGO '93 conference proceedings**

Some time ago a fellow staff member asked me if I would write a set of tools on number theory that he could use in his VCE classes, tools that would:

a) generate Fibonacci numbers,
b) test a number for primality,
c) generate the prime factors of a number d) solve pairs of linear simultaneous equations

The following paper is a 'walkthrough' of the process I followed in producing them, together with some thoughts that may be of interest to teachers.

Before finally starting I went back to the person requesting the tools and after further discussions the range was extended to the following modules:

1 Fibonacci Numbers
2 Divisors (Factors)
3 Greatest Common Divisor
4 Lowest Common Multiple
5 Prime Numbers
6 Prime Factors
7 Linear Simultaneous Equations
8 a Menu

I always begin a project like this one by consulting the literature as widely as I can and collecting ideas that I think will be useful, and as this is done for my own purposes I'm often careless about noting the source of the ideas.

T his paper is a reconstruction of that process and whilst I believe that I've referenced all my sources, it is likely that some have been missed...mea culpa,...

The code that goes with each module is at the end of the paper and is heavily internally documented.

Module1:FibonacciNumbers
The purpose of this tool is to be able to generate the requested number of members of the series known as the Fibonacci Series.
This series of numbers, which begins:

0 1 1 2 3 5 8 13 21 34

can be defined as:

$$\text{Fib}(n) = \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ \text{Fib}(n-1) + \text{Fib}(n-2) & \text{otherwise} \end{cases}$$

My first attempt at writing a procedure to generate this series was based directly upon the series of values listed above, rather than the formal definition of the series. I felt that it was so very similar to generating factorials using a one line recursive procedure that it would be easy to use this as a model. Put simply, I failed and was forced to find and use a formal definition of the series (e.g. Abelson & Sussman, 1986, p 35) to work from. The source that I used contained a procedure, written in Scheme, that I converted to Logo. It worked fine, except that it was so very, very slow that it would have driven the students crackers waiting for the numbers to appear[1] - requesting 10 values took 3 seconds, asking for 15 values required 26 seconds, 20 values took five minutes!

[1] Timing was done on a Mac Plus with 2.5 [illegible]s of RAM using LogoWriter

The solution, again found in Abelson & Sussman (page 36), made use of what they describe an iterative process and in this case the first 20 values appeared in 5 seconds. The trade-off is that this second, more efficient method, has code that is more difficult for the student to follow, as the following comparison shows:

Simple Recursive Method:

```
To Fib : number
IF : number = O [OP 0 ]
IF :number = 1 [OP 1]
OP (Fib : number Ä 2 ) + (Fib :
number-1 )
END
```

Iterative Method:

```
To Fib2 : a : b : count
IF (:count = 0) [STOP]
INSERT : a TAB
Fib2 :b (:a + :b) (:count Ä 1)
END
```

This pattern of trading speed for complexity of logic was one that was to appear regularly throughout the development of these tools.

One unintended outcome of this module is that after the 48th term, the numerical output changes from integer representation to real number representation. This provoked several worthwhile discussions with students concerning the representation of numbers within the computer and the loss of accuracy (and thus of information) that results.

Module2:GreatestCommonDivrisor

Methods for determining the Greatest Common Divisor (gcd) either by hand or using a computer are readily available; many make use of the Euclidean Algorithm. See (Courant & Robbins, 1941, p 44) and (Graham, Knuth & Patashnik, 1991, plO3), for a description of the algorithm itself; (Dromey, 1982, p97) and (Rohl, 1984, p4) provide typical implementations using Pascal whilst (Abelson & Sussman, 1985, p44) provides an implementation written in Scheme.

As my starting point I used the description provided by Graham et.al. (p. 103) which states that:

gcd (n,m) = gcd (m mod n, n)

Working through the algorithm showed exactly how it works and allowed me to develop a mental picture of the process, something that I need to have before I generate the code.

eg. gcd (18,30) = gcd (12,18)
gcd (12,18) = gcd (6,12)
gcd (6,12) = gcd (0,6)

since the last iteration produced zero as the remainder, the gcd (18,30) is 6.[2]

This can be implemented directly in Logo as:

```
To gcd : numl : num2
I F : numl = O [ OP : num2 ]
gcd (REMAINDER : num2 : numl ) : numl
END
```

a function that uses tail recursion to find the gcd of the two input values. (For the Pascalians amongst you, the I.ogo primitive REMAINDER is equivalent to Pascal's MOD)

As an aside it was instructive to compare the complexity of Dromey's analysis (some eight pages), with the simplicity of say Rohl's (at about eight lines) as something to be wary of when searching the literature for an appropriate algorithm. This became important later when considering prime number generation and prime factors - see modules 4 to 6 later.

Module3:LeastCommonMultiple

When reading Graham et. al. for ideas on determining the gcd of two integers, they referred on page 103 to exercise 2 on page 144 .

Its solution provided a simple algorithm for determining the Least Common Multiple (lcm) of two integers:

gcd (m,n) x lcm (m,n) = m x n

which gives, after rearranging:

lcm (m,n) = m x n / gcd (m,n)

As the previous module had involved developing a procedure to determine the gcd of two integers,

2 This result can be verified easily by listing all the factors (divisors) of both 18 and 30 and selecting the largest value that is common to both. This is the process that students are introduced to, but for VCE students I felt that Euclid's Algorithm was more appropriate, as well as more efficient.

the procedures for determining the lcm of two integers became obvious:

```
TO lcm :num1 :num2
OP (:num1 * :num2) / (gcd :num1 :num2)
END
```

together with

```
TO gcd :num1 :num2
IF :num1 = 0 [OP :num2]
OP gcd (REMAINDER :num2 :num1) :num1
END
```

A good example of serendipitous outcomes that sometimes occur from literature surveys particularly helpful for a relative non-mathematician like myself.

Modules 4, 5 tr 6: Factors, Prime Numbers and Prime Factors

I've decided to comment on these three modules together because their development became so intertwined.

I started by looking at the task of generating prime numbers and although I was vaguely aware that I was heading into some heavy computing and that I would need to brush-up my understanding of number theory, I really didn't expect it to be so difficult to generate prime numbers up to say 10,000. A number that seemed to me to be rather small as numbers go!

As always, I started off by developing a mental picture of what I was trying to do and to this end I readily found information on prime numbers (Courant & Robbins, p. 21) and on the Sieve of Eratosthenes (ibid. p. 25); algorithms coded in Pascal (Dromey, 1982, p. 105) were found and, in a recently purchased text by Cuoco (Cuoco, 1990, Ch 10), a series of procedures in Logo. These procedures made use of the Filter procedure (Harvey, 1985, p. 217) and gave me a way of avoiding the use of arrays and using lists as an input stream to then filter out factors and so leave only non-factors (primes) in the list.

Almost immediately problems began to appear. Whilst the method described in Cuoco worked, it resulted in the error message 'Out of stack space' for numbers very much smaller (maximum of 538 on a 2.5 Mbyte RAM Macintosh) than I was aiming for. The initial culprit was a procedure (function) called List . Upto (CUOCO, p. 276):

```
To List.Upto :num
  IF :num = 1 [OP []]
  OP SE (List.Upto :num - 1)  :num
  END
```

A quick examination showed that all invocations remain 'alive' until: num = 1 is reached, so I replaced it with Make . List, which is rather less elegant but produced larger lists and had the added advantage of being much faster:

```
TO Make.List :value :newllst
IF :value = 1 [OP []]
MAKE "newlist FPUT :value :newlist
OP MakeList :value - 1 :newlist
END
```

Make.List will handle values in excess of 10,000, modifying it to exclude even numbers greater than '2' would double this.

Having made this change I was somewhat disappointed to find the same error message for values greater than 800; this time in the procedure called Not _ a _ Factor?. Closer examination revealed that the real problem lay not with Not _ a _ Factor? itself, but with the combination of procedures used to implement the sieve. All of the central procedures use embedded recursion, each taking up stack space until the limit is reached. The only solution was to find an alternative way of generating prime numbers. Did a faster test exist that I could use in a similar manner?

An examination of the literature soon showed that a vast array of prime number generators existed ABelson & Sussman, p. 45],[Knuth, 1981, pp], [Cormen, Leiserson & Rivest, 1991, p.837] and [Ribenboim, 1991], finding one that I could decipher, let alone one that the students could understand, proved more difficult. The modern, (supposedly faster) algorithms use probabilistic methods to achieve their results, and whilst the Miller-Rabin test [Cormen et al, p. 840] is described as fast, I decided to use a test due to Wilson [Crandall & Colgrove, 1986, p. 46] which was much easier to understand and code. It turned out to be slower than the current method.

Wilson's Theorem itself says that 'p' is a prime number if

$$(p - 1)! \bmod p = p - 1$$

This can be written as a pair of Logo procedures:

```
TO Wilson? :num
MAKE "factorial 1
IF ((Wilson2 :num :factorial 1) + 1) = :num
                              [OP "true]
OP "False
END

TO Wilson2 :num :factorial :count
IF :count = :num [OP :factorial]
MAKE ~'factorial (REMAINDER (:count * :
                              factorial) :num)
OP Wilson2 :num :factorial (:count + 1)
END
```

which in turn can be incorporated into the primelIst procedure to give:

```
TO PrimeListW :list
; [this takes a list as input and deletes all non-
                              prime members]
IF EMPTYP :list [OP []]
IF (wilson? (FIRST :list)) [OP FPUT (FIRST
        :list) PrimeListW BF :list]
OP PrimeListW BF :list
END
```

Unfortunately this method is significantly slower than the method in the code provided. In Object Logo it will provide output to input numbers in excess of 5000, provided you are prepared to wait!

The code listed (based heavily on Cuoco) makes use of the original procedures and the first screen warns the user that the maximum value is 800.

Module7:LinearSimultaneousEquations

One of the simplest algorithm that can be used to solve linear simultaneous equations in two unknowns makes use of matrices and Kramer's Rule. It is also a method that the students will have used in their mathematics classes.

For two equations, say:

$$ax + by = k1$$
$$cx + dy = k2$$

in matrix form look like:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} k1 \\ k2 \end{bmatrix}$$

which will give as our solution:

$$x = \frac{\begin{vmatrix} k1 & b \\ k2 & d \end{vmatrix}}{\begin{vmatrix} a & b \\ c & d \end{vmatrix}} \qquad y = \frac{\begin{vmatrix} a & k1 \\ c & k2 \end{vmatrix}}{\begin{vmatrix} a & b \\ c & d \end{vmatrix}}$$

The coded solution is little more than the conversion of the above into Logo and in the form presented is a nightmare of FIRST and BF's, LAST and BL's. It does however, have the advantage of being very simple to follow at the level of its logic if not of its code.

```
To SimultSolv :coeffs :outlist
MAKE "x Determinant (LAST FIRST :coeffs)
(LAST LAST :coeffs) (FIRST BF FIRST :coeffs)
                        (FIRST BF LAST :coeffs)
MAKE "y Determinant (FIRST FIRST :coeffs)
            (FIRST LAST :coeffs) (LAST FIRST
                  :coeffs) (LAST LAST :coeffs)
MAKE "det Determinant (FIRST FIRST :coeffs)
        (FIRST LAST :coeffs) (FIRST BF FIRST
              :coeffs) (FIRST BF LAST :coeffs)
MAKE "outlist LIST (: x / :det) (:y / :det)
OP :outlist
END


TO Determinant :RlC1 :RlC2 :R2C1 :R2C2
OP (:RlC1 * :R2C2) - (:R2C1 * :RlC2)
END
```

Module8:Menu

The tools presented here have been field tested by a year 7 mathematics class and all the modules work quite well except for a 'fault' present in all of them. When prompted for an input, be it a single number or a yes/no, the modules do not check adequately for a legitimate response, instead they accept what they are given and then 'crash' when it is the wrong type. This would need to be remedied if you wanted to usc it regularly in a class.

This problem can be easily overcome by use of some error trapping routines that check to see if the response is appropriate and beeping or whatever if it is not. These programming requirements are not peculiar to Logo but common to all languages.

Code1:FibonacciNumbers

```
TO ; :comment
END

TO Fibonacci
CT HT
PR [This program prints out as many Fibonacci
numbers as you request.]
PR [To use it, press the <RETURN> key and
enter the number of terms required. It assumes
that ZERO is the first term]
Pause
Fib
END

TO Pause
; [cause execution to stop until a key is pressed]
IF NOT EMPTY? READCHAR [STOP]
Pause
END

TO Fib
; [the 'main' program, prompts for how many]
; [and then calls Fib2 to generate them]
CT
BLines 3
(INSERT [How many terms?:] Blank)
MAKE "number FIRST READLIST
BLines 2
IF :number = O [PR 0 STOP]
Fib2 0 1 :number
PR Blank Blines 2
Repeat? [Generate more? (Y/N)]
END

To Fib2 :a :b :count
; [outputs the fibonacci numbers, :count is a
                    decrementing counter]
IF (:count = 0) [STOP]
INSERT :a TAB
Fib2 :b (:a + :b) (:count Ä 1)
END


TO BLines :num
; [prints :num blank lines]
REPEAT :num [PR []]
END

TO Blank
; [prints a blank character]
OP CHAR 32
END

TO Repeat? :message
; [prompts to see if further fibonacci numbers
are wanted]
Blines 1 TAB
INSERT :message
MAKE "ans FIRST READLIST
  IFELSE OR (:ans = "Y)(:ans = "y) [CT
Fib][Finish]
END

TO Finish
CT
BLines 9 TAB TAB PR [Goodbye.]
WAIT 30
CT CLEARNAMES
GETPAGE "Menu
END
```

Code2:GreatestCommonDivisor

```
TO ; :comment
END

TO BLines :num
REPEAT :num [PR []]
END


TO GCDivisor
CT CG HT
PR [This program determines the Greatest]
PR [Common Divisor (GCD) of two integers.]
GCD.Input
END


TO GCD.Input
BLines 2
 INSERT [Enter the two numbers whose GCD
you wish to find: ]
MAKE "Numbers READLIST
MAKE "A FIRST :Numbers
MAKE "B LAST :Numbers
BLines 2
 (INSERT [The GCD of] blank :A blank [&]
blank :B blank [is:] blank)
PR GCDiv :A :B
BLines 2
Repeat? [Solve another? (Y/N)]
END


TO GCDiv :numl :num2
  ; [determines the greatest common divisor of
:numl and :num2]
IF :numl = O [OP :num2]
OP GCDiv (REMAINDER :num2 :numl) :numl
; if :numl > :num2 then the first time through
they simply get]
  [ reversed, hence input order does not matter]
END
```

```
TO Repeat? :message
; [prints request for more, if not then 'Finish']
INSERT :message
MAKE "ans FIRST READLIST
IFELSE OR (:ans = "Y)(:ans = "y) [CT
GCD.Input] [Finish]
; [anything other than 'y' or 'Y' results in
'Finish']
END

TO Finish
; [cleans up and returns to the main menu]
CT BLines 9 TAB TAB
PR [Goodbye.]
WAIT 30
CT CLEARNAMES
GETPAGE "MENU
END

TO Blank
; [prints a blank character]
OP Char 32
END
```

Code3:LeastCommonMultiple

```
TO ; :comment
END

TO Blank
OP CHAR 32
END

TO BLines :num
REPEAT :num [PR []]
END

TO LowComMult
CT HT
PR [This tool determines the LCM of two
integers.]
PR [It makes use of the fact that the LCM of
two numbers A & B is given by]
PR [A x B/ GCD(A,B).] LCM.Input
END

TO LCM.Input
PR [] PR []
(INSERT [Enter the two numbers whose LCM is
to be determined:] Blank)
MAKE "numbers READLIST
MAKE "numl FIRST :numbers
MAKE "num2 LAST :numbers
PR []
(INSERT [The LCM of] :numl [&] :num2 [is:]
blank )
PR LCMult :numl :num2
PR []
Repeat? [Solve again? (Y/N):]
END

TO LCMult :numl :num2
OP (:numl * :num2)/(GCDiv :numl :num2)
END

TO GCDiv :numl :num2
IF :numl = O [OP :num2]
OP GCDiv (REMAINDER :num2 :numl) :numl
END

TO Repeat? :message
INSERT :message
MAKE "ans FIRST READLIST
IFELSE OR (:ans = "Y)(:ans = "y) [CT
LCM.Input][Finish]
END

TO Finish
CT
BLines 9
TAB TAB
PR [Goodbye.]
WAIT 30
CT CLEARNAMES
GETPAGE "MENU
END
```

Code4:Divisors(Factors)

```
TO Divis? : num : denom
; [returns 'True' if num/denom = integer]
OP (REMAINDER : num : denom) = O
END

TO MakeList :value :outlist
IF :value = O [OP :outlist]
MAKE "outlist FPUT :value :outlist
OP MakeList :value Ä 1 :outlist
END

TO Apply :function :value
; [used in the procedure Filter]
; [combines a function and a value and then
RUNs it]
OP RUN SE :function :value
END

TO Divisors.Calc :number
; [applies a division 'filter' to the list of
integers up to :number]
; [i.e. outputs those numbers in the list that
divide into :number]
OP Filter "Divis? :number (MakeList :number
[])
END
```

```
TO Filter :pred :value :inlist
 ; [returns a list of value that have been filtered
according to]
 ; [the predicate test that has been applied to each
value in :inlist]
 ; [the empty list becomes the 'first' member of the
list of divisors]
IF EMPTY? :inlist [OP[]]
IF (Apply :pred SE :value (FIRST :inlist))
   [OP FPUT (FIRST :inlist) (Filter :pred :value
(BF :inlist))]
OP Filter :pred :value (BF :inlist)
END

TO SumList :inlist
IF empty? :inlist [OP 0]
OP (FIRST :inlist) + SumList BF :inlist
END


TO Divisors
CT HT
PR [This tool determines the divisors]
 PR [of a number and their sum and prints them
out.]
Divisors.Input
END


TO Divisors.Input
BLines 2
(INSERT [Enter the number whose divisors are to be
found:] Blank)
MAKE "number FIRST READLIST
BLines 1
(PR [The divisors of] :number [are:])
Blines 1
MAKE "divlist Divisors.Calc :number pr :divlist
BLines 1
(PR [and their sum is:] sumlist :divlist)
Repeat? [Solve another? (Y/N)]
END


TO Blank
 OP CHAR 32
 END


TO BLines :num
REPEAT :num [PR []]
 END


TO Repeat? :message
Blines 1
TAB INSERT :message
MAKE "ans FIRST READLIST
IFELSE OR (:ans = "Y)(:ans = "y)  [CT
Divisors.Input][Finish]
END


TO Finish
CT BLines 9
TAB TAB PR [Goodbye.]
WAIT 30
CT CLEARNAMES
GETPAGE "MENU
END
```

Code5:PrimeNumbers

```
TO ; :comment
END

TO PrimeNumbers
CT HT
PR [This program determines and prints out the
prime numbers up to the value entered. The maximum
value is 800]
Prime.Calc
END


TO Prime.Calc
 INSERT [Enter the upper limit:]
 MAKE "value FIRST READLIST
CT (PR [The prime numbers less than] :value [are:])
 BLines 1
 Display Prime_Upto_E  :value
 PR "
 Repeat? [Print to a new maximum? (Y/N)]
 END


TO Display :inlist
; [prints out a list of values as a simple table].
IF EMPTY? :inlist [STOP]
(INSERT FIRST :inlist " )
Display BF :inlist
END


TO Prime? :number
; [tests to see if :number is a prime number]
IF :number = 2 [OP "True]
IF Even? :number [OP "False] ; [if its even then not
prime]
OP PrimeHelper :number 3 ; [first divisor is 3 - odd
numbers only]
END

TO PrimeHelper :num :divisor
IF (:divisor > sqrt :num) [OP "True]
; [no need to check beyond ~num]
IF Divis? :num :divisor [OP "False]
OP PrimeHelper :num (:divisor + 2)
END

TO Divis? :numerator :denominator
; [returns 'True' if]
OP (REMAINDER :numerator :denominator) = 0
END
```

```
TO Even? :num
OP Divis? :num 2
END

TO PrimeList :list
; [a prime number filter that works by testing each
member of the]
; [list for primality it results in a list of primes]
IF EMPTY? :list [OP []]
IF Prime? (FIRST :list) [OP FPUT (FIRST :list)
PrimeList BF :list]
OP PrimeList BF :list
END

TO MakeList :value :newlist
; [clumsy, but in tail recursive form to handle larger
numbers]
; []
IF :value = 1 [OP :newlist]
MAKE "newlist FPUT :value :newlist
OP MakeList :value - 1 :newlist
END

TO Not_A_Factor? :f :n
OP NOT Divis? :n :f
END

TO Eratosthenes :inlist :outlist
;[this works by progressively removing (sieving) all
of the ]
;[multiples of values < the SQRT of the largest from
:inlist]
;[This leaves two lists of primes, one the values
smaller than ]
;[SQRT of largest and another of the larger values ]
;[ -they are then combined as the final step]
; []
IF (FIRST :inlist) > SQRT (LAST :inlist) [OP
                        SE :outlist :inlist]
OP Eratosthenes (SIEVE "Not_A_Factor? (FIRST
    :inlist) BF :inlist) (LPUT FIRST :inlist :outlist)
END

TO Primes_Upto_E :number
; [the Eratosthenes sieve version]
OP Eratosthenes MakeList :number [] []
END

TO Sieve :pred :value :inlist
; [see comments in Eratosthenes above]
IF EMPTY? :inlist [OP []]
IF Apply :pred (SE :value FIRST :inlist) [OP FPUT
        FIRST :inlist (Sieve :pred :value BF :inlist)]
OP Sieve :Dred :value BF :inlist
END

    TO Apply :function :value
    ; [this
    OP RUN SE :function :value
    END

    TO Repeat? :message
    Blines 1
    INSERT :message
    MAKE "ans FIRST READLIST
      IFELSE OR (:ans = "Y)(:ans = "y) [CT
Prime.Calc][Finish]
    END

    TO Finish
    CT
    BLines 9
    TAB TAB
    PR [Goodbye.]
    WAIT 30
    CT CLEARNAMES
    GETPAGE "MENU
    END

    TO BLines :num
    REPEAT :num [PR []]
    END
```

Code6:PrimeFactors

```
TO ; :comment end

TO PrimeFactors
CT
PR [This program determines the prime factors of
a number and prints]
PR [them out.]
PR [It works for numbers up to at least 100000
but can be slow!]
PrimeFactors.Input
END

TO PrimeFactors.Input
BLines 2
INSERT [What number do you wish to find the
prime factors of?]
REPEAT 2 [INSERT Blank]
MAKE "number FIRST READLIST
BLines 2
(PR [The prime factors of] :number [are:])
BLines 1
TAB
Factors :number []
Repeat? [Solve another? (Y/N)]
END

To Prime? :number
; [tests to see if :number is prime Ä slow!]
IF :number = 2 [OP "True]
IF Even? :number [OP "False] ; [if its even its
not prime]
OP PrimeHelper :number 3
END
```

```
TO PrimeHelper :number :odd_integer
; [reduces the numbers to be tested by excluding all even numbers>2] ]
IF (:odd integer > SQRT :number) [OP "True]
; [no need to check beyond sqrt :num]
IF Divis? :number :odd_integer [OP "False]
OP PrimeHelper :number (:odd_integer + 2)
END


TO Divis? :numerator :denominator
; [returns 'True' if num/denom = integer]
OP (REMAINDER :numerator :denominator) = O
END


TO Even? :num
OP Divis? :num 2 ; [op's "true" if :num is even]
END


TO MakeList :value :newlist
; [inelegant but memory efficient list maker]
IF :value = 1 [OP :newlist]
MAKE "newlist FPUT :value :newlist
OP MakeList :value Ä 1 :newlist
END


TO Not_A Factor? :f :n
OP NOT Divis? :n :f
END


TO Eratosthenes :inlist :outlist
IF (FIRST :inlist) > sqrt (LAST :inlist) [OP SE :outlist :inlist]
OP Eratosthenes (SIEVE "Not_A_Factor? (FIRST :inlist) BF :inlist) (LPUT FIRST :inlist :outlist)
END


TO Primes_Upto_E :number
OP Eratosthenes MakeList :number [] []
END


TO Sieve :pred :value :inlist
IF EMPTY? :inlist [OP []]
IF Apply :pred (SE :value FIRST :inlist)
        [OP FPUT FIRST :inlist (Sieve :pred :value BF :inlist)l
OP Sieve :pred :value BF :inlist
END


TO Apply :function :value
OP RUN SE :function :value
END


TO Factor_List :number :prime_list
; [determines which of the numbers in the list of primes]
; [is a factor of :number]
IF EMPTY? :prime_list [OP []]
IFELSE Divis? :number FIRST :prime_list
        [OP FPUT (SE FIRST :prime list Ord FIRST :prime list :number)
Factor_List :number BF :prime_list]
      [OP Factor_List :number BF :prime_list]
END


TO Ord :prime_no :number
; [this op's the no of times that :prime no divides into :number]
IF NOT Divis? :number :prime_no [OP 0]
OP 1 + Ord :prime_no (:number / :prime_no)
END


TO Factors :number :PList
; [this prints the prime factors of a number]
IF Prime? :number [PR (SE :number [is a prime number]) STOP]
; [checks to see if the number is prime]
MAKE "PList (Factor List :number Primes Upto E INT SQRT :number)
; []
IFELSE ((Factor_Check :PList) = :number)
[Display :PList STOP]
[Display LPUT (SE (:number / Factor Check :PList) 1) :PList]
END


TO Factor_Check :Factor_List
; [this multiplies out and sums the sets of factors in :Factor_List]
IF EMPTY? :Factor List [OP 1]
OP (Mult_Out FIRST :Factor_List) * Factor_Check BF :Factor_List
END


TO Mult_Out :input
; [takes a list of two numbers and outputs the first raised to the second]
OP PowerM FIRST :input LAST :input
END


TO PowerM :base :index
; [op's :base raised to :index]
IF :index = O [OP 1]
OP :base * PowerM :base (:index Ä 1)
END


TO Display :PList
; [displays the list of factors]
IF EMPTY? :PList [CB DELETE PR " STOP]
REPEAT LAST FIRST :PList[INSERT FIRST FIRST :PList INSERT CHAR 46]
Display BF :PList
END
```

```
To Blank
OP CHAR 32
END

TO BLines :num
REPEAT :num [PR []]
END

TO Repeat? :message
Blines 1
TAB INSERT :message
MAKE "ans FIRST READLIST
IFELSE OR (: ans = "Yl ( ~n.~; = "vl r~
END

TO Finish
CT
BLines 9
TAB TAB PR [Goodbye.]
WAIT 30
CT CLEARNAMES
GETPAGE "MENU
END
```

Code7:LineatSimultaneousEquations

```
To ; :comment
END

TO SimEqu
SimultEqu []
END

TO SimultEqu :outlist
CT HT
PR [This tool will find the solution to a pair of
linear simultaneous equations.]
PR [of the form aX + bY = K] BLine 1
PR [When asked to enter the coefficients, they
must be entered in the following order:]
PR [ <x-coeff> <y-coeff> and <constant>]
PR [ for each equation.]
BLine 1
PR [Enter coefficients for first equation:]
MAKE "XYK1 READLIST
PR [Enter coefficients for next equation:]
MAKE "XYK2 READLIST MAKE "coeffs LIST
:XYK1 :XYK2
CT
PR [Your equations are:] TAB (INSERT FIRST
FIRST :coeffs "x "+ FIRST BF FIRST :coeffs "y
"= LAST FIRST :coeffs)
PR " TAB (INSERT FIRST LAST :coeffs "x "+
FIRST BF LAST :coeffs "y "= LAST LAST :coeffs)
BLine 2
PR [The solutions are:] MAKE "Ans SimultSolv
:coeffs [] TAB TAB (INSERT [x =] Char 32)
PR FIRST :Ans TAB TAB (INSERT [y =] Char
32)
PR LAST :Ans Repeat? [Solve another? (Y/N)]
END

To SimultSolv :coeffs :outlist
; [uses FIRST, BF, LAST & BL to extract relevant
values from the lists]
; [really only a 'trial' solution that needs to be tidied
up and made]
; [much more modular instead of having all the
MAKE's] 3
MAKE "x Determinant LAST FIRST :coeffs LAST
LAST :coeffs FIRST BF FIRST
    :coeffs FIRST BF LAST :coeffs
MAKE "y Determinant FIRST FIRST :coeffs FIRST
LAST :coeffs LAST FIRST
    :coeffs LAST LAST :coeffs
MAKE "det Determinant FIRST FIRST :coeffs FIRST
LAST :coeffs FIRST BF
    FIRST :coeffs FIRST BF LAST :coeffs
MAKE "outlist LIST (:x / :det)(:y / :det)
OP :outlist
END

TO Determinant :RlC1 :RlC2 :R2Cl :R2C2
; [evaluates a 2 x 2 determinant]
OP (:RlC1 * :R2C2) Ä (:R2C1 * :RlC2)
END

TO Repeat? :message
BLine 1 TAB INSERT :message
MAKE "ans FIRST READLIST
IFELSE OR (:ans = "Y) (:ans = "y) [CT
SimEqu][Finish]
END

TO BLine :num
REPEAT :num [PR []]
END

TO Finish
CT
BLine 9
TAB TAB PR [Goodbye.]
WAIT 30
CT CLEARNAMES
GETPAGE "MENU
END
```

Code 8: Menu

```
TO STARTUP
CT CC CG HT
NumberStuff
END

To ; :message
END
```

3.See Lewis Ch 4 for more information

```
TO NumberStuff
  ; [this is the main menu that controls access to]
  ; [all of the 'tools']
    ; [it uses multiple IF's rather than a custom written]
    ; [case statement simply to make it easier for students to read]
  ; []
  HT CT
  BLine 1
 PR
[**************************************]
INSERT [*]Blanks 28 INSERT [Number Theory Menu] Blanks 29 PR [*]
 PR
[**************************************]
  BLine 1
   TAB PR [1 -Fibonacci Numbers]
   TAB PR [2 - Prime Numbers]
   TAB PR [3 - Prime Factors]
   TAB PR [4 - Divisors]
   TAB PR [5 -Greatest Common Divisor]
   TAB PR [6 - Least Common Multiple]
   TAB PR [7 - Simultaneous Equations]
   BLine 1
   TAB PR [9 QUIT]
   BLine 1
   TAB INSERT [Select by typing a number:]
   MAKE "select FIRST READLIST
   Selection :select
   END
   TO Selection :select
    ; [this procedure calls the relevant page and runs it]
   ; []
 IF :select = 1 [CT CC Fibonacci]
 IF :select = 2 [CT CC PrimeNumbers]
 IF :select = 3 [CT CC PrimeFactors]
 IF :select = 4 [CT CC Divisors]
 IF :select = 5 [CT  CC GCD]
 IF :select = 6 [CT CC LCM]
 IF :select = 7 [CT CC SimultEquations]
 IF :select = 9 [CT CC QUIT]
 CT
 BLine 10
 PR [Numbers between 1 & 9 only please!]
 NUMBERSTUFF
 END
```

```
TO Fibonacci
GETPAGE "FIBONACCI
RUN "Fibonacci
END

TO PrimeNumbers
GETPAGE "PRIMENUMBERS
RUN "PrimeNumbers
END

 TO PrimeFactors
GETPAGE "PRIMEFACTORS
 RUN "PrimeFactors
 END

 TO Divisors
 GETPAGE "DIVISORS
 RUN "Divisors
 END

 TO GCD
 GETPAGE "GCDIVISOR
 RUN "GCDivisor
 END

 TO LCM
 GETPAGE "LCMULTIPLE
 RUN "LowComMult
 END

 TO SimultEquations
 GETPAGE "SIMULTEQUS
 RUN "SimEqu
 END

 TO Blanks :n
 ; [prints multiple spaces]
 REPEAT :n [INSERT Blank]
 END

 TO Blank
 ; [prints a space]
 OP CHAR 32
 END

 TO BLine :num
 ; [prints a blank line]
 Repeat :num [PR []]
 END
```

Bibliography

Abelson, H. and Sussman, G. J., *Structure and Interpretation of Computer Programs*, MIT Press, Cambridge, 1985.

Cormen, T. H., Leiserson, C. E. and Rivest, R. L., *Introduction to Algorithms*, MIT Press, Cambridge, 1 990.

Courant, R. and Robbins, H., *What is Mathematics? An Elementary Approach to Ideas and Methods*, Oxford U. P., London, 1941.

Crandall, R. E. and Colgrove, M. M., *Scientific Programming with Macintosh Pascal*, John Wiley & Sons, New York, 1986.

Cuoco, A., *Investigations in Algebra*, MIT Press, Cambridge, 1990.

Dromey, R. G., *How to Solve It by Computer*, Prentice-Hall International, I.ondon, 198Z.

Graham, R. L., Knuth, D. E. and Patashnik, O., *Concrete Mathematics*, Addison Wesley Publishing Co, Reading, 1989.

Harvey, B., *Computer Science Logo Style*, MIT Press, Cambridge, 1985.

Knuth, D. E.,*The Art of Programming; vol 2 SemiNumerical Algorithms*, AddisonWesley, Reading, 1968, second edition 1981.

I.ewis, 1'.1)., *Approaching Precalculus Mathematics Discreetly*. MIT Press, Cambridge, 1990

Ribenboim, 1'. *The Little Book of Big Primes*, Springer-Verlag, New York, 1991.

Rohl, J. S., *Recursion via Pascal*, Cambridge U. P., Cambridge, 1984.

Other books that may bc of interest include:

Boecker, H-E., Eden, H. & Fischer, G.,*Interactive Problem Solving Using Logo*, Lawrence Erlbaum Associates, NewJersey, 1991

Burke, M. P. & Genise, L. R., *Logo and Models of Computation*, Addison-Wesley, 1987

Eisenberg, M., Clinger, W., & Hartheimer, A., *Programming in MacScheme*, MIT Press, Cambridge, 1990 (a PC version is also available)

Friendly, M., *Advanced Logo,* Lawrence Erlbaum Associates, NewJersey, 1988

Springer, G, & Friedman, D. P., *Scheme and the Art of Programming*, MIT Press, Cambridge, 1989

DELTA: Design Environment for Learning Top-Down design for Algorithms

Trevor Barrett

(This article is based on a workshop at ACCE '93 about DELTA, by C.J.Webb, which was presented by Trevor Barrett.)

Introduction

The purpose of this session is to demonstrate/ examine a software package called **DELTA,** which is currently being used in various schools throughout Queensland in the teaching of the unit **Algorithms and Programming** in the senior subject **Information Processing and Technology**.

The acronym DELTA stands for Design Environment for Learning Top-down design of Algorithms, and was developed by C.J.Webb of Griffith University, Queensland. DELTA is a C.A.S.E. tool to assist in the development of an algorithm for the solution to a problem, and the later translation of that algorithm to an appropriate language. The program runs on most I.B.M. compatible computers, and while a mouse is not essential, it does simplify the use of the software.

During this presentation about DELTA, my main aim is to attempt to convey that this software package is a very useful solution to various problems which can be encountered in the teaching of a unit on Algorithms and Programming.

In the teaching of such a unit, I have frequently found that students and teachers can tend to view the coding aspect af the unit as its primary aim, and rather than conveying principles of algorithm design, the unit becomes an exercise in learning about Turbo Pascal or some other programming language.

Indeed, it was not uncommon for me to see the 'formal' algorithm design process occur after the coding had taken place, so that the flowchart, the Nassi-Shneiderman diagram, pseudocode or other algorithm representation methodology matched he code which had already been produced.

In using DELTA to develop an algorithm, the student is 'forced' to focus on a problem solving plan, such as that devised by the mathematician Polya i.e.

Define -> Devise -> Test -> Revise

These problems can be expanded as follows:-

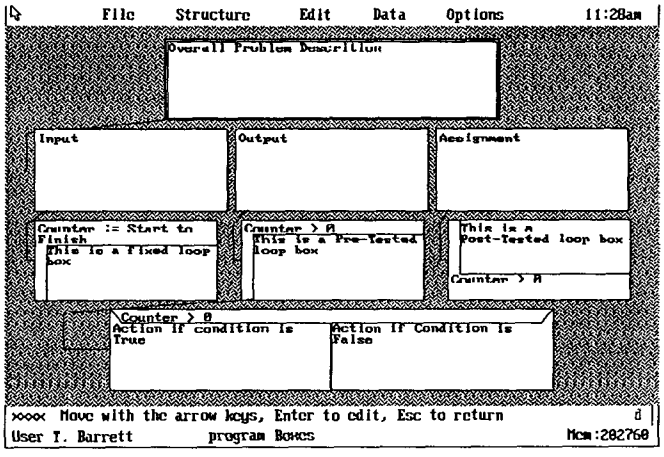
(a) Define - the specification of the problem in a form suitable to a computer solution, and even to consider if the computer solution is appropriate in the circumstances. An ill-defined problem can produce a similar solution. Generally speaking, problems that are suited to a 'computer solution' are those where an answer (output) has to be produced, after some data (input) has been supplied, and the necessary processing is a task suited to the computer.

(b) Devise - the development of the solution to the problem - the algorithm. In the development of this phase, DELTA uses Structure Design Charts, a combination of Nassi-Shneiderman diagrams and tree diagrams. The algorithm for the solution to the problem is presented in a graphical, easy to understand form.

A Top-Down design process is employed in DELTA. As well, the software provides for Djikstra's theorem which demonstrates that any problem that is solvable using a computer can be specified under constructs of sequence, selection, and iteration.

The problem is first defined in general terms, and the student is then required to progressively break down the task into smaller and smaller sub-tasks. In so doing, the student must consider whether they are dealing with concepts of input, output, assignment, a module, selection, or iteration.

The following diagram illustrates the symbols that are used in DELTA to represent various programming constructs.



The rectangles are used to represent problem descriptions, or processes of input, output, or assignment. There are also boxes for iteration (fixed, pre-tested, and post-tested loops) and levels of selection.

I would point out here that use of the software does, in my opinion, require some lead-up work with the students. For example, students must understand the constructs mentioned previously, be familiar with a variety of data types suited to various types of information, and appreciate the concept of a variable. (c) Test - DELTA provides a 'trace' facility which enables the student to test run the program, to see that the algorithm does in fact produce the outcome that was expected. This occurs still within the DELTA environment, and at this stage no coding into any computer language has occurred.

(d) Revise - If the solution does not behave as expected, the algorithm can be modified, and retested until a correct solution is found.

To me, the beauty of such an approach is that students are not restricted by, or required to learn, the constraints of the syntax and operation of any particular computer language. The emphasis remains on the problem solving process, rather than the use of whatever language is chosen to implement the solution.

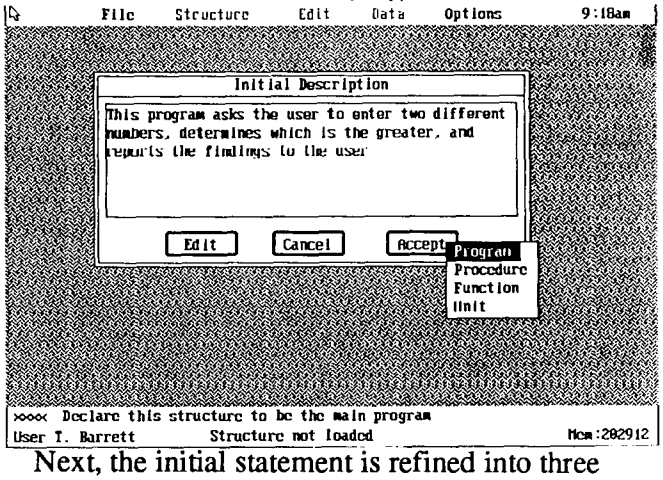
This fact that coding is only one phase of the entire process is really brought home to the students when they first observe that their solution, in diagrammatic form, can easily, and automatically encoded by DELTA into a variety of languages, currently including Pascal, Turbo Pascal, C, Modula2, Paradox, and Open Access.

On the following pages a simple problem is developed, and then there are various exercises which workshop participants might wish to attempt using the software.

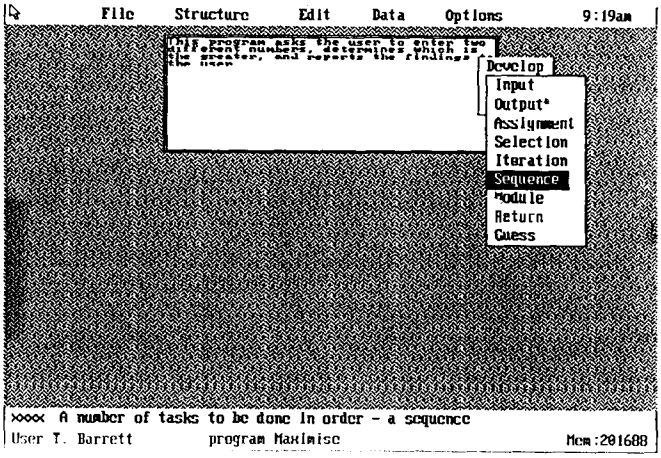
Sample Problem

The problem under consideration is a simple one, namely to ask the user to enter two different numbers, determine the greater number of the two entered, and report the findings to the user.

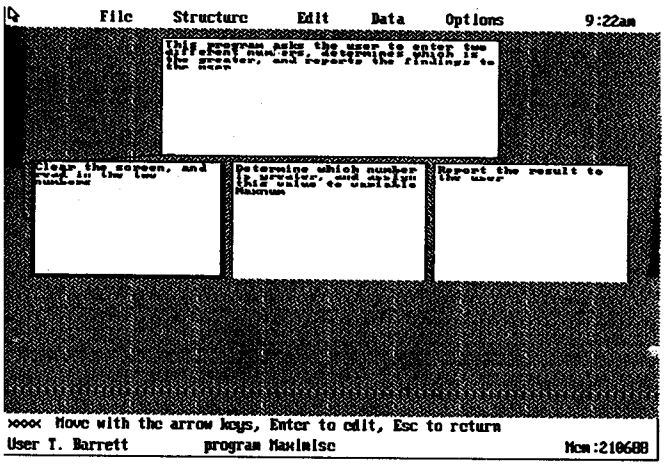
To begin, **New** is selected from the **File** menu, and the name *Ages* is entered. Then a general statement of the problem is given, and the problem declared to be a program.



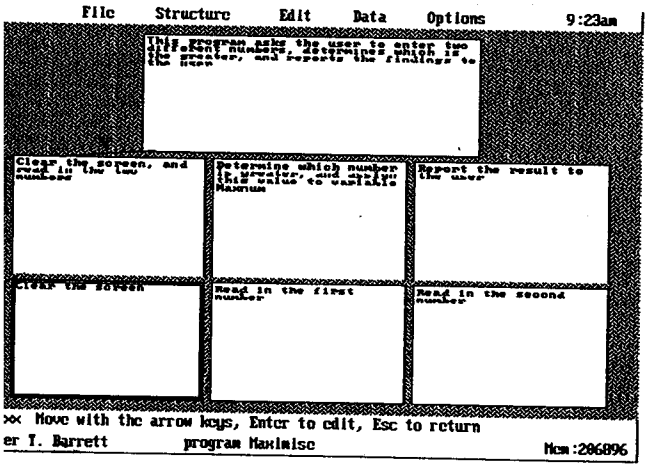
Next, the initial statement is refined into three sub-tasks, by choosing **Edit**, then **Develop**, then **Sequence** from the menus which are offered.



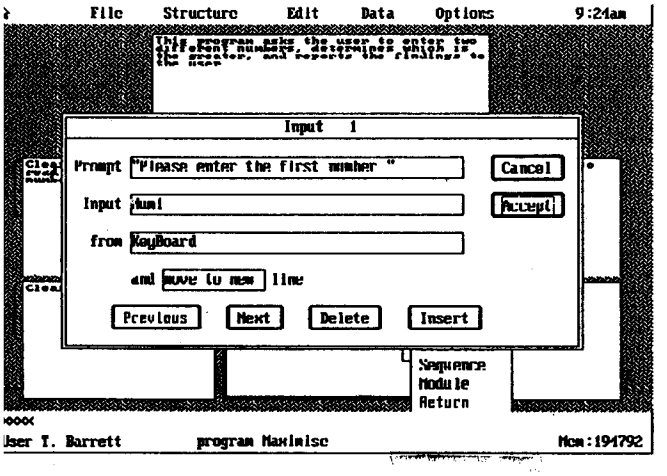
The three **sub-tasks** are outlined as follows;



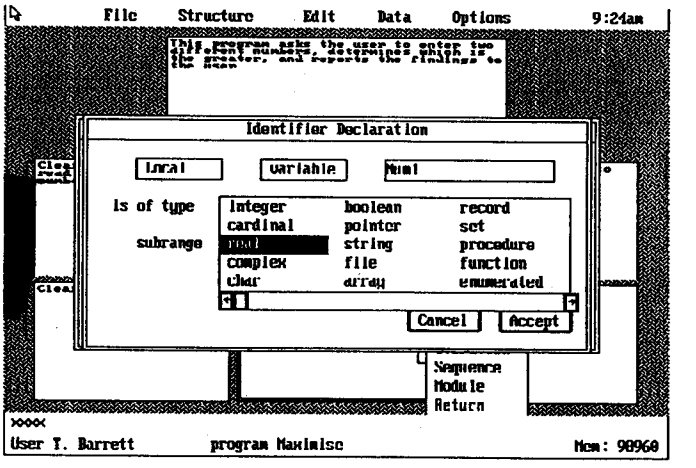
The first box is then further developed by a similar process, into a **sequence** of three further sub-tasks



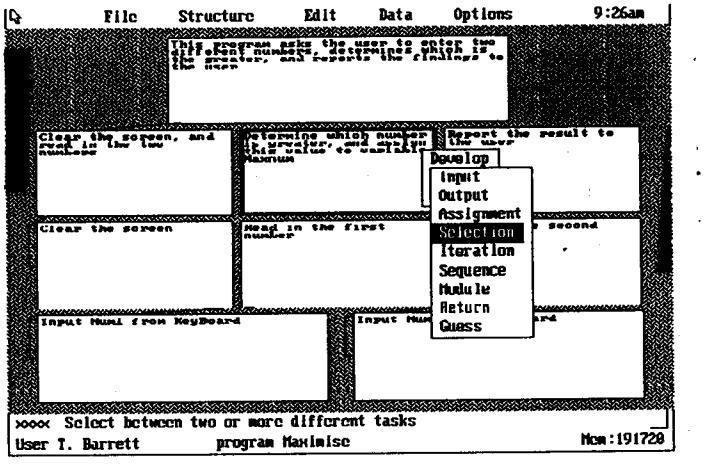
The box to **clear the screen** will be developed later. At this stage, the box to **read in the first number** is developed as an **Input**



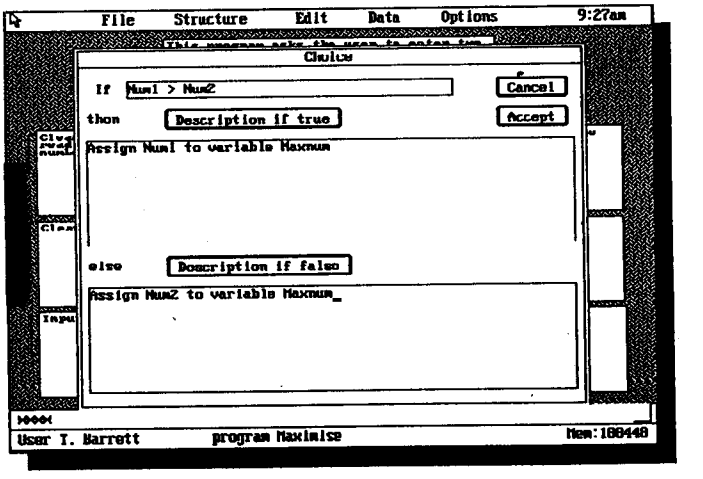
DELTA detects the variable **Num1** and details about this variable are entered. The subsequent box to **read in the second number** is developed in a similar way



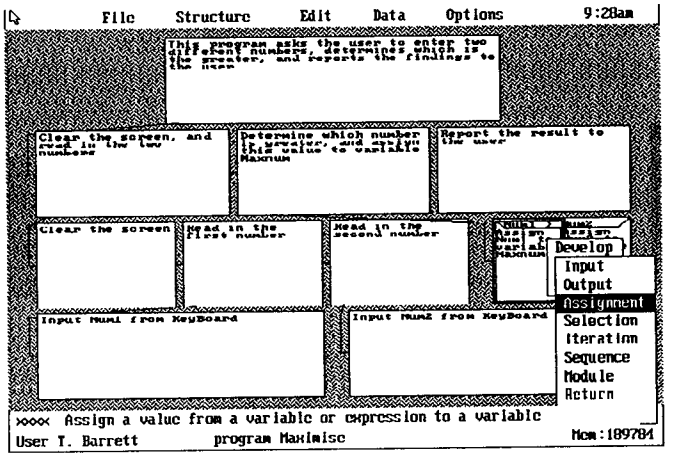
The next step is to develop the box which **determines which number is greater.** This is done as a **Selection**

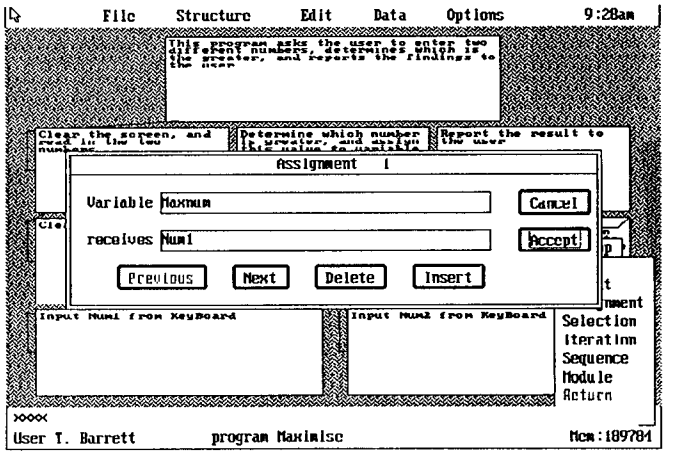


The resulting **If-Then-Else** construct is described as follows

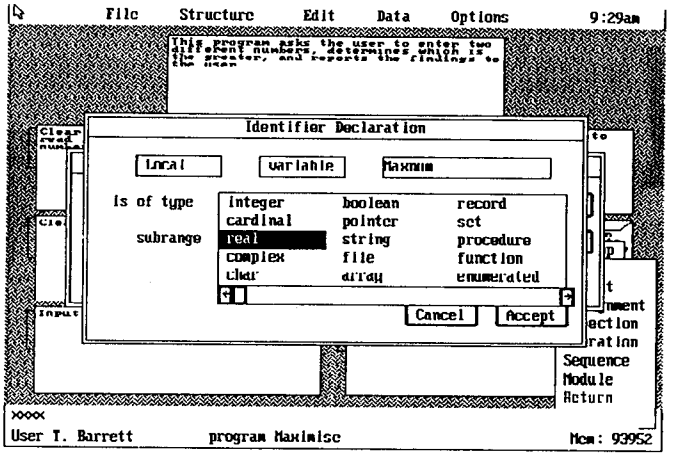


The left hand side of the Selection box is developed as an **Assignment** operation, as outlined in the following two diagrams.

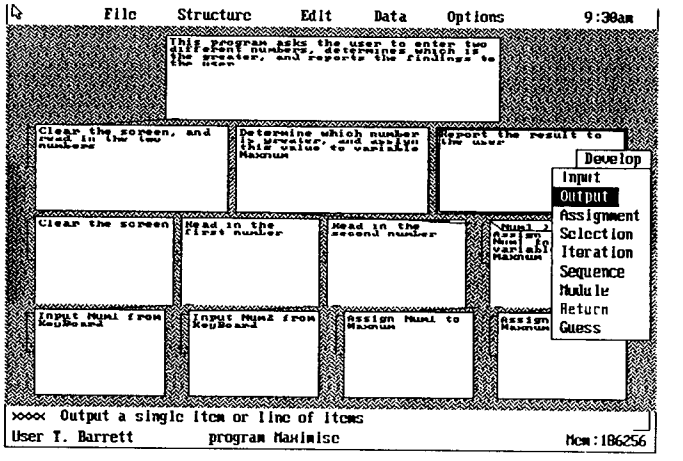




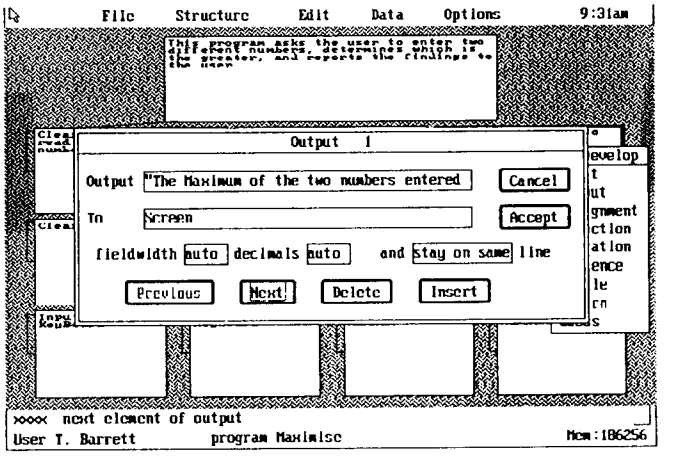
DELTA detects the new variable **Maxnum**, and details about this variable are entered. The right hand side of the Selection box is developed in a similar way



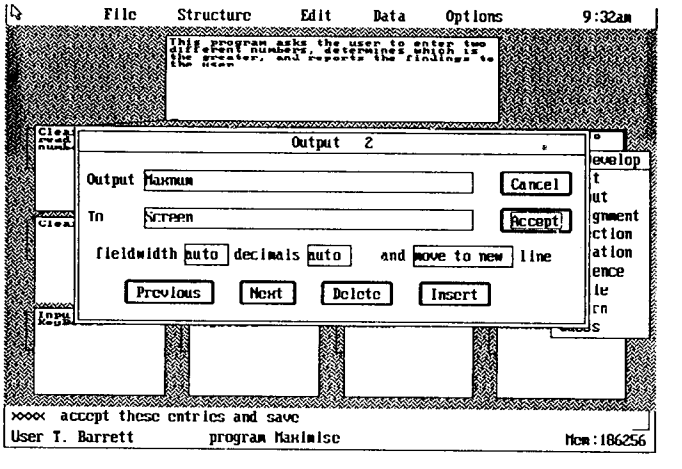
The next step is to develop the Output phase of the program, as shown below



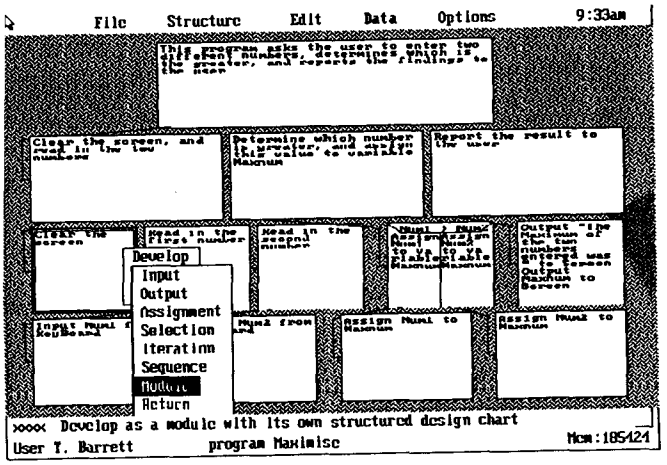
The output is developed in two stages. The first of these is a **string** of data, as a lead-in to the maximum value.



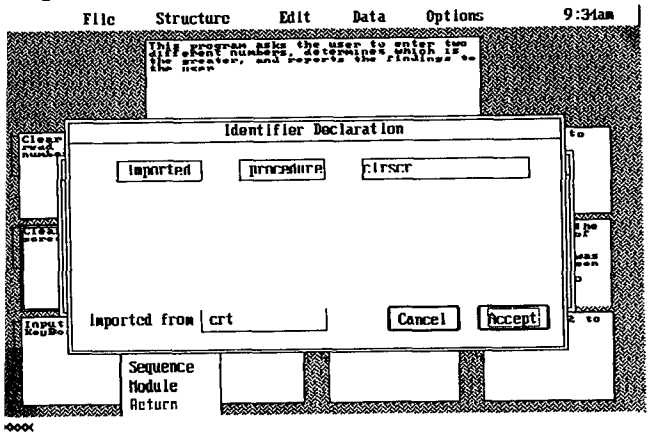
The second output screen contains the actual variable to be output. Note that this value will follow the introductory phrase



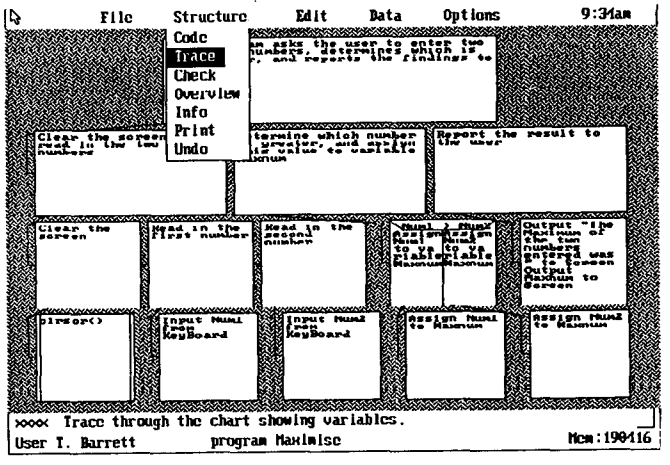
Returning to the **clear screen** option which was by-passed earlier, this can be developed as a **module** (as 'clearing the screen' is actually a procedure)



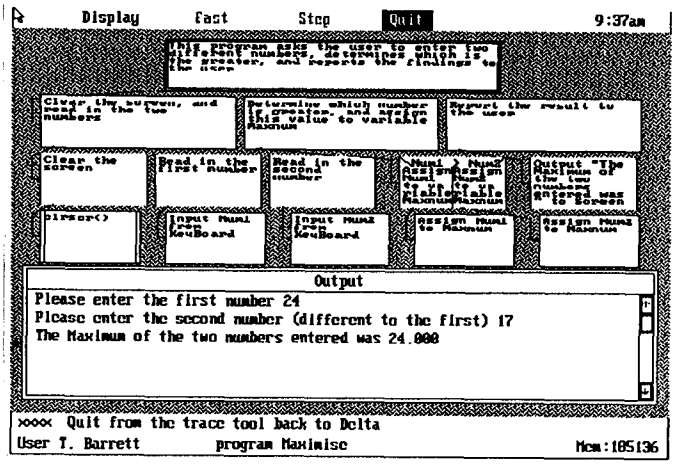
We declare the module **clrscr** to be an imported procedure from the **unit CRT**



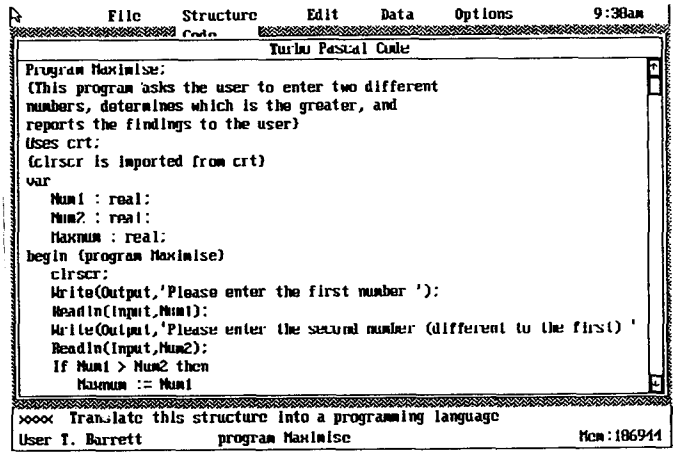
We can now **test** the algorithm by choosing the **Trace** option from the **Structure Menu**. Trace can be run in Step-Mode, or Fast-Mode



The assignment of values to variables, and output from the trace can be followed on screen. If necessary, the Structured Design Chart can be modified, and Trace re-run until the output is as required.



Finally, the Structured Design Chart can be **translated** into the chosen language, in this case Turbo Pascal. The file is saved to disk with the extension **.PAS**



Obviously, we have only had a brief look at the DELTA software. There are many other aspects of the software which can be explored, including the development of modules (user defined procedures and functions, and imported modules), built in functions, arrays, records, files, etc.

From a teaching point of view, one useful facility is the **Playback** option in the **File** menu. This enables a teacher to prepare a problem beforehand, and then *playback* the steps in the development of the Structure Design Chart.

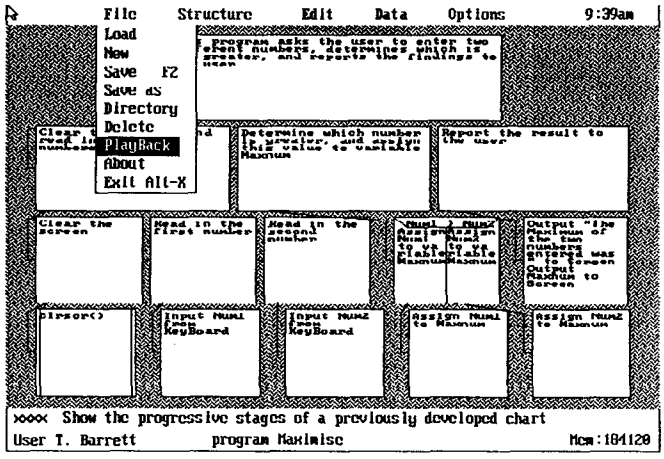
Example Task:
Use DELTA in the development of solutions to the following problems:-

1. Read in your birth, the current year, and determine your age in years.

2. Read in two numbers from the keyboard, and determine their average.

3. Write a program to convert degrees Fahrenheit to degrees Celsius (9C = 5F - 160)

4. Write a program to read in a number and determine the square root of the number. (SQRT is a built-in function in Turbo Pascal)



A new version of the software is under development, and this is hoped to eliminate some possible problems with the current version. For example, the Trace facility cannot be used with Functions.
However, as a teaching tool, I believe the software to be particularly useful.
Any persons requiring further information on the package should contact:

Structured Solutions
P.O. Box 859
Mt. GRAVATT
Qld 4122
Ph: (07) 849 2404

Acknowledgements

Webb C.J. (1992) 'Structured Design Charts and DELTA', QSITE State Conference Proceedings 1992 pp105-115

Gordon C. (1993) 'An introduction to DELTA', QSITE - IPT Teachers Conference

Carey D. (1993) 'Advanced DELTA Techniques', QSITE -IPT Teachers Conference)

The Use Of CD-ROM Systems

Leonie Fraser
Miller Educational Technology Centre

Why have CD ROM drives become popular?

CD ROM is an innovation that is primarily industry driven - in response to the requirements set down by the users of software. A recurrent theme in the criticism of software has been that the users of computers want better and more pictures and sound, with programs that give more information, in depth, with greater teacher control.

There is a limit to the amount of information that can be put onto a floppy disc. The number of programs that are installed onto a hard disk to run, because they have been compressed to fit onto floppy discs for purchasing is growing. A CD or Compact Disc can store a much larger file, of around 600 Mg (megabytes) of information. This means that there is room for the size of files people want to use, with pictures, sounds and animations, as well as movies.

For those with a mind to know, a byte of information represents a character - a letter, number, or any other key press at the keyboard (such as a space or return). A kilobyte is one thousand bytes. It usually takes about 2k (the symbol for kilobyte) to store a page of typing. It is not unusual for a graphic (digital picture) to take up to 500k of memory space. One megabyte is one thousand kilobytes. It takes approximately 1mg (the symbol for megabyte) to store 1.6 minutes of digitised sounds. An animation with many pictures and sounds takes up a large amount of memory. Some software developers, such as Microsoft, have started putting their applications programs (wordprocessors, databases or spreadsheets, for example) together on CD, so that they give users of their programs extensive tutorials and a database style manual that allows searching to be done for any word at any location in the manual.

Another reason software is being developed in this format is that CD's protect the copyright of the software company. The program on a CD cannot be easily copied to another disc. CD's are inexpensive to produce, and are therefore easily upgraded. The life expectancy of a CD (treated well) is also much higher than a more traditional disc.

Using a CD ROM system.

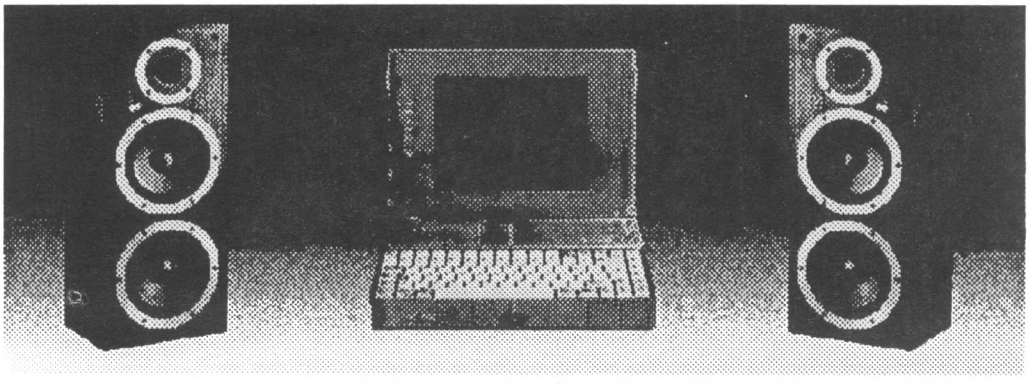
Some CD ROM systems require the installation of software supplied with the CD ROM Drive into the computer's operating system so it can recognise that a drive exists. CD ROM Drives plug into the computer via a SCSI lead and card. Some CD's will also contain files to be copied onto the computers hard drive which will then run from the hard drive and read the CD ROM drive at appropriate times. Without these files installed on the computer, the CD will not run.

CD ROM systems are available for all major education suppliers. All offer a range of similar features and their own unique features which make them all valuable and viable systems. ACORN The video handling program for the Acorn CD system is called Replay. At present, Replay allows video segments to be shown in full screen size at 18.5 frames per second, or at 25 frames per second at quarter screen size. The components of Acorn CD's, such as The **Dictionary of the Living World**, can be saved as text, picture, video, and sound files by students or teachers who can then use the information to construct their own presentations using multimedia programs, such as **Magpie**, **Genesis** or **Omnis**. Acorn do not make a monochrome system, so all presentations fully support colour.

The hardware to run the range of Acorn CD's would include the entire range of their computers - including the non-hard drive systems. The best presentations are seen on the A3020, A440, A540 or A4000 and A5000. These machines, with 4Mb

of RAM are configured for quality presentation of all Acorn CD's. The monitor supplied with the A4000, A5000 and A540 systems is a high resolution, multisync monitor, so the visual resolution is high quality. Multisync monitors are available for the A3020 as an option. As well as running RISC OS Acorn computers will also run MS DOS with a software emulation, and, through an emulation card, Windows 3.1, and so will run DOS CDs and some Windows CDs.

AMIGA Commodore run the CDTV systems with a hand controller similar to their game controllers. There is a huge range of 'game' type programs available for this system, some of which have existed in the education market in disc form for years. The system requires a player similar in appearance to an audio CD player or a video



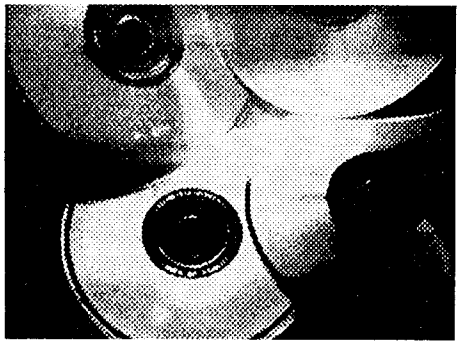
player which includes dedicated computer processing and a television screen. The CDTV Player can be outfitted as a computer, by adding a monitor, keyboard and mouse.

DOS There is a large range of DOS CD's on the market. CD's running under DOS will not include video or large amounts of animation, although they may include pictures and sound. Generally they have a large amount of textual information. Newer CD's may run under the Windows operating system. Many Windows CDs include animation and video clips, as well as sound files. The basic DOS machine may not run a CD straight away when a player and SCSI card are installed into the machine. For most new CDs including animation and video, a Super VGA or Multisync monitor is required, with a Super VGA video card replacing the existing video card in the computer. Some CDs will also require the computer have a Sound card installed, and may require that the computer have more than 4mg of RAM to give sufficient free space once Windows is running to control the CD. This problem can be addressed by using memory controlling software, such as that included in DOS 6. Video on DOS requires either the multimedia extensions developed by Microsoft, or Quick Time for DOS.

MACINTOSH There is a growing range of Australian curriculum applicable Macintosh CD's on the market. Some CD's relying solely on Hypercard as the controlling program do not yet run in colour, or may not run in colour on all machines. Others which have been developed in MacroMind, or other animation authoring programs, run best in colour or only in colour. The Macintosh video handling program is called Quick Time and runs in a window approximately 10cm by 7cm on the screen. The quality and resolution of this tends to be very grainy, with a tendency for the words and faces to appear out of synchronisation if the CD is running on a low end Macintosh such as an LCII. Best performance is achieved by running the CD on a top end of the range Macintosh - such as a VI, VX, Centris or Quadra. The frame speed is 10 frames per second.

The minimum system for running CD's is 4Mb, however the speed is vastly improved by running on a machine with a larger RAM. A high resolution multisync monitor, such as those Apple has begun to sell with their new systems, also improves the quality of the image considerably. All systems need a stereo speaker system to handle sound well. The computer can handle the functions of an amplifier in a sound system, but not that of speakers. The internal computer speakers will not handle the excellent music which is a feature of CD presentations. Some systems include this in the purchase price of the CD player.

Macintosh and Amiga systems are dedicated. The drives will not run discs configured for other platforms. The Acorn and DOS players are interchangeable. DOS CDs and Amiga CDTV CDs can be made to run on an Acorn with specially written 'front end' software.
To use the CD players currently available for computer systems, the user must place the CD in a caddy, with the label side up. The caddy system is an obvious marker to the user that they are using a Computer CD player, as the case resembles a 3.5" floppy disc case. The real difference is in the way in which the drive scans for information on the disc; for audio systems the choice is between sampling the signal 4 or 8 times. For Computer CDs the signal is sampled 32 times - a much higher quality machine mechanically.



The difference between the audio and computer systems using and not using caddys has caused some problems, as the caddys are expensive to purchase alone and add to the cost of the system unnecessarily. The manufacturers have said that future computer CD players will not use caddys.

As far as the user of the CD is concerned, the drive will appear and be addressed in the manner usual to that system - on Macintosh and Amiga, there is no icon when the drive is empty. Once the disc has been inserted, the icon appears as another disc on the desktop which is opened by double-clicking, programs run similarly, quit by pressing Open Apple/Q and ejected by Trashing the icon.

Acorn CD drive icons appear in the Icon bar at boot up and are opened by clicking on once, closed by closing the window and ejected manually. The drive can be dismounted if you are swapping through a number of CD's.

Under Windows, an Icon will be placed on the desktop when the installation program for the CD is run. Under DOS, selecting the CD drive in the usual manner (eg, F:), then browsing the directory of the CD will show which is the exe file to run the CD.

The CD's and their Contents

CD's are in their infancy - there is a growing number and a great deal of variation in the quality. Just as with books and other software items, teachers must evaluate them carefully and decide on their usefulness. Teachers more familiar with computer presentations will look past the massive leap in the capabilities of the technology to the actual quality of the product. CD's that allow the reading level of the text to be set at a more appropriate level for the ability of the students are going to be more appropriate than CD's that simply reproduce the text in an existing encyclopaedia. Problems that can be noted in books and software with cultural bias, sexism, inaccurate information and inappropriate content are all also problems with CD's.

It is also necessary to assess the interactivity of the presentation style, and the amount of use the system can reasonably take in a school.

CD's are essentially information based. Many schools are seeing them as a library appropriate purchase rather than a faculty or class based purchase. After all, the CD encyclopaedia are cheaper and more easily upgraded than the book versions - as well as taking up less space. In a library, if one, two or three students only are using the CD ROM machine, what do the rest of the library users access? The long term plans could include a number of machines all including CD ROM Drives - as common in a library as the shelves that hold the books are today perhaps.

These are a sample of the CDs currently available, with some comments on them as program examples. As with all software, the content of CDs will change and expand as the medium is explored and becomes more common in schools.

The Parliament Stack - First appeared as a program on disc. This program is distributed through the Education Office of Parliament House, Canberra. It is a hypercard stack containing information on all the electorates in Australia. The files include Census information,

electoral office information and demographics, and pictures of the representative and where they are located when parliament is sitting. The discs are updated every time there is a bi - election, and the updated CD is available as an upgrade on an exchange and small fee basis. This is a good example of CD's providing a large amount of detailed information that can be easily updated.

Creepy Crawlies - This is a CD featuring insects - and creepy crawly thingies - much loved by children! The interesting feature of this reference book is that the textual information is included on the CD four times - in two different reading levels and two different languages. Younger students, or less able readers, can select a stage that allows them to access the information in a more appropriate format. The CD also has a picture and video file - in colour, for each crawly shown.

Grandma and Me - this is the warm fuzzy of CD programs at present - the most reluctant computer user will happily sit and click with a happy smile on their face to discover all the hidden animations in the program. The CD contains an animated story that the reader or listener can explore. There is no feedback from the reader incorporated into the story, so it is still very like the encyclopaedia type of CD.

Revelation - this CD incorporates a program that has been available on disc for some time - with a very impressive collection of files that students can use with the program, or in other programs. **Revelation** is a professional quality Art program, and the files included on the CD provide a library of photographs, pictures and paintings that have been digitised for student use. Students can manipulate the colours in a Van Gough painting of sunflowers, for example, and discover what colour the sunflowers really are. The CD Storage space has been used to best advantage through the inclusion of an interactive tutorial program.

Animals - this CD presents a visit to the San Diego Zoo, where the user can choose from a map which biome they will visit. Going to the area shows them information, pictures, audio tape and video about the landscape and the creatures in it. The whole is icon driven, and incredibly easy to navigate. A strong feature is the video on the endangered species program running at the zoo, which is over 5 minutes in length. Parts of this video can be quickly accessed when searching the information on an individual animal, thus providing a very useful video cross-reference index in the background programming.

Beethoven's 5th - this is an example of a CD created by the music industry having the blank space on the disc after the audio material has been burnt on used for a computer control program. The small space taken by the program is enough to allow the user to learn about the period in which the music was written, the music itself, and musical terms, phrases and methods as the 5th Symphony illustrates. Also included is a music quiz game. All aspects search and find the appropriate pieces of the music in the CD to illustrate the point being covered in the tutorial program.

What will the future hold?

While prophesizing is not one of my strong points, there have been several indications that this technology will have some dramatic contributions over the next few years. Indications are that more interactive programs are due to be released, where students will make a response at an appropriate point in the program that will be used to channel the student to an appropriate next stage or section of the program. Resource material and syllabus material could well be delivered to teachers via this medium. Software developers are working to develop systems that utilise the medium in new ways. Students can now explore the Antarctic with Scott on the CDTV system, reading about the expedition and trying to manage their own in a successful manner. This type of simulation is ideal as the CDs hold so much varied information. Kodak CDs have become available, and some schools are already exploring putting commonly used and valuable pictures on CD for use in school publications. Clip art collections in this genre are sure to become popular. Acorn, for example, already provide a clip art collection of useful video clips. There are also many collections of graphic design clip art CDs for the Macintosh. One thing is certain, CD programs have the potential to deliver increasing variations in software over the next few years.

Pacific PowerNET :A resource for teachers.

John Attwood
McCarthy Catholic Senior High
Tamworth

I was privileged to represent the Yr 11/12 SIG of the CEG, and the Schools of the North West, at the recent launch of an exciting new electronic communications network, called **Pacific PowerNET.**

PowerNET was officially launched by Premier, John Fahey, at an event held at Mt Piper Power Station on Friday, September 3rd. Mr Fahey was accompanied by Mrs Virginia Chadwick (in her capacity as Minister for Education) and Mr Garry West (Minister for Energy), as he initiated a phone call which sent messages of congratulations to his two colleagues.

Following this, the visitors saw a demonstration of PowerNET in use by several Year 12 students from the Calrossy School (Tamworth). They have been involved in trialling the network since the regional hub was established at the school earlier this year.

Sponsored by Pacific Power (the electricity generation and management utility of NSW), this network is of immense interest to teachers at all levels of the education process. Of particular interest to members of the Yr 11/12 SIG is the low cost, hands-on availability of on-line databases, echo-mail and electronic mail facilities.

The basis of the network is the establishment of a Fidonet (an international, low-cost network) hub in Sydney. Regional hubs, like the one established at Calrossy, will progressively be established at schools in strategic locations across the state. Other schools can then call into the network for the cost of calling the regional hub school. The central hub will periodically update the regional hub data bases and will effect transfer of messages to/from the regional hubs (at no cost to the regional site) and will conduct the international message transfers.

At its very simplest, this setup will allow Computing Studies students to gain practical experience at establishing and carrying on echo-mail and e-mail communications.

Echo mail can be used by students for simple "chat" with people in other countries. This has implications for many subject areas. One example from the Calrossy experience will demonstrate the enormous potential.

A student, required to present a Personal Interest Project for assessment in Society and Culture, placed an echo-mail message requesting information/views relating to women in politics. Within days, messages from U.S.A., Canada and other countries flowed in, allowing her project to be completed with an international flavour, and showing some real insights into the place of women in politics world-wide. In addition, the student was able to send e-mail messages to US President, Clinton, using the White House e-mail address.

[illegible] information that was available on Viatel (forerunner to Discovery), yet the major difficulties with Viatel, the high cost and slow response times, are largely overcome using PowerNET.

As well as the data bases, and the availability of e-mail and echo-mail, there are areas available for games, downloadable software, and more features are in planning.

For teachers of any subject, at any level, theres an echo-mail area for you! As well as the obligatory computer literacy and computing studies areas, there are areas relating to mathematics, geography, languages (some in English, others in the language), science, the environment, cookery.. to name but a few that spring to mind.

As computer educators, you owe it to your colleagues to at least let them see the potential of this form of communication presents. More to the point, you owe it to your students to allow them access to the combined wisdom of a few thousand educators, and their peers, on the PowerNET network. You should start lobbying for a regional hub in reasonable range of your school, so you can be a part of this low-cost, high-interest, forum for educators and their students.

Your professional development is also catered for, with international and local teacher "chat" areas available. You can establish contact with peers in Europe, Canada, Asia, New Zealand, and, of course, U.S.A.. In this way you can keep a handle on the emerging issues from overseas in classroom management and/or teaching strategies. You can also be a part of the discussion!

All it takes is a simple phone call to your regional hub twice a week (or more often, if you wish) and downloading the areas you have an interest in. Then these messages are read off-line, comments or responses are composed offline and uploaded the next time you log on. Your initial logon requires you to answer some very innocuous questions about yourself, for future identification purposes. This process can be completed in about five minutes, at the [illegible]. Future logons will identify you by name and a private password, which you can change.

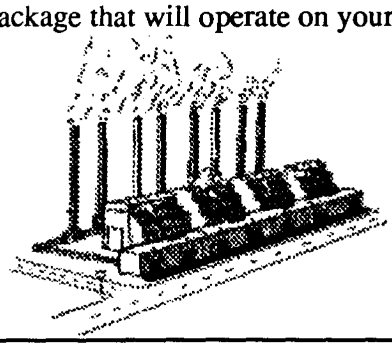
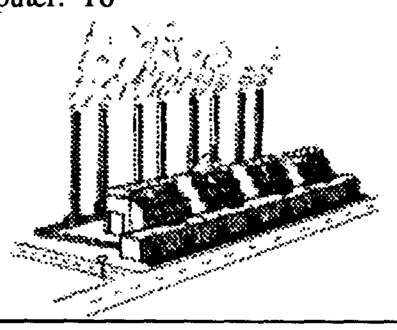
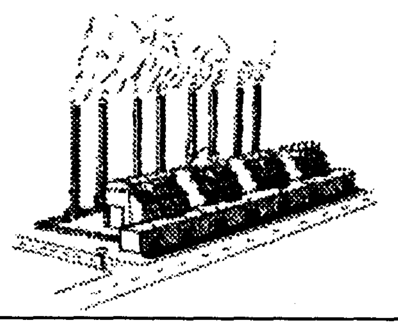
Frequently asked questions (FAQs) at the launch concerned how much the access was going to cost, or how much equipment/software was required. The answer, for most users, is a resounding "VERY LITTLE!!". If you don't already have a computer at the school, then you will, of course, need to get one that is capable of operating a modem (and that includes just about all of them!). You will also need a modem and phone access. You'll also need a communications package that will operate on your computer. To allow for off-line reading of the messages and responding to them, you will also need an off-line reader package capable of reading QWK formatted and compressed files. For MS-DOS computers, the comms package could be as expensive as you like, but I have found that the shareware packages are the equal of most commercial efforts. I use Telix by preference, but Procomm plus works just as well (both are shareware).

Off-line readers are predominantly shareware too. I use the OLX (for Off Line eXpress) reader, but any of the Silly Little Mail Reader (SLMR) family will work well. Again, both of these mentioned are shareware, and should be available at a BBS near you (if one exists!) but I can supply a copy of Telix and OLX on one High Density disk for $5 to cover the costs of duplication and postage (MS-DOS only).

I am not all that familiar with the equivalent packages for Macs, but I know they do exist. From the references I have seen, Freddie and MacWoof are the Mac off-line readers of choice at present. Search around your local Mac shareware dealers, they might have these, or others that will [illegible] reasonable [illegible]. One caveat! The reader [illegible] ding [illegible] repacking the replies into the same [illegible].

If you need help with establishing a setup, I'd be happy to [illegible] if I can, as will the people at Pacific Power. C[illegible] School on: (067) 618676, or Pacific Power's Community Relations at the following numbers: (02) 268 6800 OR (008) 451 241. Hellen Phillips, Phil Farrell or Roger Buck can assist you.

All that remains is to mention that the ball is now squarely in your court! ... get on-line and GO FOR IT !!!!! Type To You Later (TTYL)

Review of Caddsman Cadet

by Greig Tardiani

(This is a Software Evaluation subbmitted to IT by a teacher who has used the proforma developed by the Computer Education Unit. The Evaluation is therefore done under headings that allow a thorough review. IT welcomes reviews in all formats - it is useful for other teachers to see how you rate a piece of software that they may be considering as a potential purchase. Ed.)

Title	**Caddsman Cadet**
Computer(s) for which this software is available	**IBM Clone, DOS 2.1 or higher**
Topic area(s)	**Computer Aided Drawing**

Age group	**LS, MS, US, A.**
Ability group	**All levels of secondary level**
Date/version	**1990, Version 2.12**
Author	**Cadds Man Ltd**
Publisher	**Cadds Man Ltd**
Price	**$750 for 10 machines. $1 250 for a site license**
Recommended Retail	**$7 500**
Supplier	**Cadds Man Ltd or (Davron Daizo)**
Copyright, Licensing and availability details	**Single copy and site license available. Available in either 3.5" or 5.25" disks.**

Additional software required	**DOS 2.1 or higher.**
Required hardware	**computer 286 or higher, 386 recommended.**

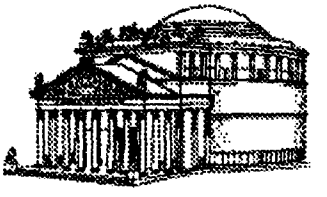
Memory	**1 Meg of Ram**
Other	**Math co-processor essential.** **3 button mouse (all three are used)**
Optional hardware	**Digitiser pallet or Puck.** **Plotter printer for output.**
Contents of Package	**Three 3.5" compressed program disks.** **Three manuals: Reference manual,** **Programming language reference manual, and** **Reference manual update for version 2.12**

Level of Evaluation

Screened	[X]
Full evaluation by subject specialist	[ ]
Field tested	[X]
Evaluated by (full name)	Greig Tardiani
Date	From 1990 to 1993
Contact point	Grantham High School
Phone	(02) 622 7300

Evaluators background and/or field test situation

The evaluator has been an Industrial Arts teacher for 14 years and computer co-ordinator for 3 years. The package has been extensively trialed with all ability levels in real life classroom situations. The test platforms were IBM 286 based computers with DOS 3.3 and 1 Meg of DRAM.



Type of software **CAD**

Description

Caddsman is a full function commercial 3D CAD program adapted with the help of well designed tutorial programs for educational use. The program has 999 layers, multiple pens, and full walk through facilities.

Educational Objectives

To expose student to an easy to use yet powerful CAD package that will expose them to the integration of computers into the areas of commercial Design and Drafting.

Method of use
Individual, Small groups, Concept development, Class, Extension and Enrichment.

EVALUATION SUMMARY

Rating Scale: **I - Impressive S - Satisfactory U - Unsatisfactory**

Content Rating [I] S U
The base program contains all the necessary drafting tools of a high-end 3 D CAD package. Icons replace menus, the program handles text well, and works with grid and co-ordinates for accurate drawing. The tutorial package that comes with the program is educationally based. An architectural package is also available.

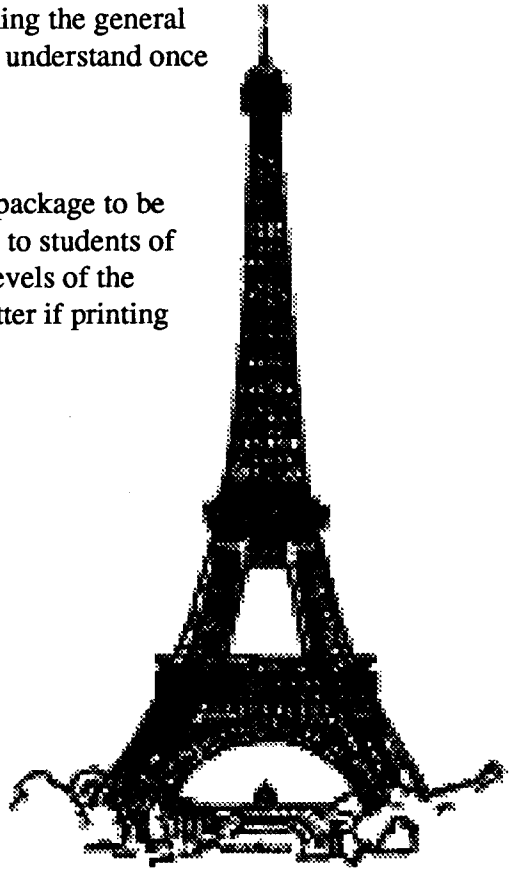
Program Features Rating [I] S U
All default values are pre-set allowing one to start drawing immediately. All three mouse buttons are used negating the need to use the keyboard. Unlike Autocad, you always work in 3D mode and can view from any angle the object being drawn.

Classroom Application Rating [I] S U
Exceptionally good CAD package for introducing high level drawing to all levels of students. The program displays how professionals can draw on a computer. Designing of projects using the tools supplied allows the student to experiment with different sizes etc. and view the project from any angle before proceeding with manufacture.

Support Material Rating [I] S U
The South Australian Department of Education have developed tutorial modules specifically for Caddsman. Technical Schools of The Future (TSOF) have developed icon packs to replace the menus and tutorial packages aimed at making learning the general program enjoyable. General manuals are well written and easy to understand once the drawing terminology is understood.

Recommendations
I would highly recommend this package as it allows a true CAD package to be used within a school teaching environment that is understandable to students of any level. It also allows the advanced student to expand to high levels of the drawing fields. One detriment to the program is the need of a plotter if printing is to be of a reasonable speed, normal printers are very slow.

Reference:
NSW Department of Schools Education
Services Directorate
Computer Education Unit
SOFTWARE EVALUATION FORM



What's Best in Business Isn't Always Best in Your Classroom

Are you truly using the BEST Educational Computers in Your Classroom?

COMPUTERS FOR THE *Curriculum*

YOU know that your classroom isn't an office, a factory, store or fast food franchise.

BUT there is a common misapprehension that the computer that's the choice of business should 'trickle down' to your classroom.

TAKE a closer look at the arguments. Is the best computer for business the best computer for the curriculum?

DOES your classroom really work in the same way as the front office, or the local grocery store? Do you need to teach which keys to press or do you prefer to teach concepts?

DOES the software you want in your classroom need to be flexible, innovative and meet the needs of students at various stages of development while being educationally significant?

*YOU **really** should ask to see the* **Acorn** *range of educational computers. Then you can make an **informed** choice between just a computer and a **Computer for the Curriculum**.*

Choose a computer for your classroom that's designed for YOUR industry... EDUCATION!

CALL Lisa on
008 032 604
for a Free
Computers for the Curriculum
Info Pack

ACORN COMPUTERS AUSTRALIA Pty Ltd
7/190 George Street, Parramatta, NSW 2155,
Phone: (02) 891 6555, Fax: (02) 635 9641
12 Gipps Street, Collingwood, VIC 3066,
Phone: (03) 419 3033, Fax: (03) 419 2892

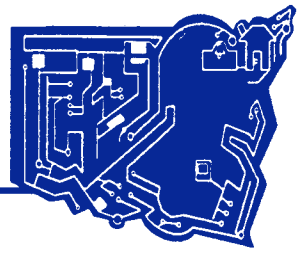
Acorn

Computers for the Curriculum

A.C.N. 006 398 205

New South Wales Computer Education Group Ltd.
Membership Form 1994

ACN 002 827 401



NSW Computer Education Group
c/- Instructional Technology Centre
School of Education
Macquarie University
NSW 2109

Phone: (02) 805 9456
Fax: (02) 805 9453
Hours of Business: 9-6 Mon-Thurs
9-1 Friday

New Member ☐ Renewal ☐ Membership Number ____

Membership for 1994 $55 ☐

Full-time Student Membership for 1994 $25 ☐
(Full time student members MUST have this form signed by a lecturer or Officer of their institution for verification)

I would like to join the following Special Interest Groups

Computing Studies 11-12 ☐ Hypermedia ☐ Early Childhood ☐

Fill in this section to provide information for our database:

1. Institution ____
2. Title ____ 3. First Name ____
4. Surname ____
5. Address Street ____
Suburb/Town ____
State ____ Post Code ____
6. Phone home ( ) ____ work ( ) ____ fax ( ) ____
7. Full-time student verification
Signature of Lecturer/Officer ____ Name of Lecturer/Officer ____

Fill in this section to provide information for mailing IT and the Newsletter:

1st line of Address (name) ____
2nd line of Address ____
3rd line of Address ____
Suburb/Town ____
State ____ Post Code ____

Throughout the year you will receive regular mailouts from the CEG. The information you enter here is exactly what will be put on your mailing label.

Fill in this section to provide payment details

Payment Method: cheque/money order ☐ credit card ☐ cash ☐

Amount: $ ____

Please debit my credit card account $ ____ for payment of 1994 NSW CEG membership

Mastercard/Bankcard/Visa Expiry Date ____

Signature ____

Members will receive a membership card which will serve as a receipt

Office Use Only

Entered on DB

Postcode Group